Interactive segmentation based on component-trees

Nicolas Passat, Benoît Naegel, François Rousseau, Mériam Koob, Jean-Louis Dietemann

Interactive segmentation based on component-trees

Nicolas Passat[a], Benoît Naegel[b], François Rousseau[a], Mériam Koob[c], Jean-Louis Dietemann[c]

[a] *Université de Strasbourg, LSIIT, UMR 7005 CNRS, France*
[b] *Université Nancy 1, LORIA, UMR CNRS 7503, France*
[c] *Université de Strasbourg, LINC, FRE 3289 CNRS, France*

Abstract

Component-trees associate to a discrete grey-level image a descriptive data structure induced by the inclusion relation between the binary components obtained at successive level-sets. This article presents an original interactive segmentation methodology based on component-trees. It consists of the extraction of a subset of the image component-tree, enabling the generation of a binary object which fits at best (with respect to the grey-level structure of the image) a given binary target selected beforehand in the image. A proof of the algorithmic efficiency of this methodological scheme is proposed. Concrete application examples on magnetic resonance imaging (MRI) data emphasise its actual computational efficiency and its usefulness for interactive segmentation of real images.

Key words: Mathematical morphology, component-tree, grey-level images, interactive segmentation.

1. Introduction

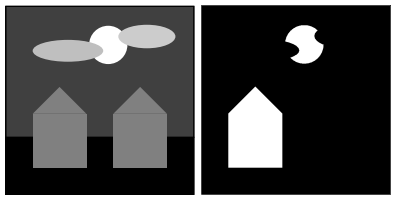
The *component-tree*, also known as *dendrone* [1, 2], *confinement tree* [3] or *max-tree* [4], is a graph-based structure which models some characteristics of a grey-level image by considering its binary level-sets obtained from successive thresholding operations. Initially proposed in the field of statistics [5, 6], the component-tree has been (re)defined in the theoretical framework of mathematical morphology and involved, in particular, in the development of morphological operators [7, 4, 8].

By definition, component-trees are particularly well-suited for the design of methods devoted to process and/or analyse grey-level images based on *a priori* hypotheses related to the topology (connectedness) and the specific intensity (locally/globally minimal or maximal) of structures of interest. (The case of colour images is currently under investigation [9, 10]; the involvement of component-trees in fuzzy grey-level images is described in [11].) Based on these properties, but also thanks to methodological developments related to complex knowledge handling [12, 13, 14], component-trees have been involved in the design of several image processing/analysis applications, especially for filtering/segmentation applications.

It has to be noticed that several works related to component-trees have been devoted to enable their efficient computation, under specific assumptions or in more general cases [3, 4, 7, 15, 16, 17, 18]. In particular, the ability to compute component-trees in (quasi-)linear time opens the way to the development of interactive and efficient segmentation methods, provided that the operations performed on an image *via* its component-tree also present a low algorithmic complexity.

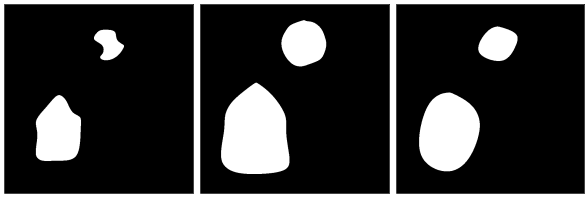
The design of interactive segmentation methods is a quite active research field. This dynamism is in particular justified by (*i*) the increasing necessity to analyse images in a large spectrum of application fields (medical imaging, remote sensing, biometry, metrology, etc.), (*ii*) the difficulty to develop fully automatic segmentation methods (parametrisation, initialisation, etc.), and (*iii*) the importance to develop segmentation methods as tools for assisting the user by explicitly using its expertise.

*Corresponding author: Nicolas Passat. Tel.: (+33) (0)3 68 85 44 96, Fax: (+33) (0)3 68 85 44 55.
Email addresses: passat@unistra.fr (Nicolas Passat), benoit.naegel@loria.fr (Benoît Naegel), rousseau@unistra.fr (François Rousseau)



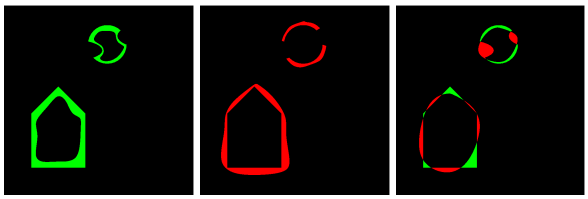
(a) (b)

Figure 1: (a) A grey-level image. (b) A segmentation of (a), assumed to be the expected one.



(a) (b) (c)

Figure 2: Three approximate segmentations of Figure 1(a): (a) undersegmentation, (b) oversegmentation, and (c) mixed under/oversegmentation, with respect to the expected result of Figure 1(b).



(a) (b) (c)

Figure 3: The errors of the segmentations of Figure 2 (false positives in red, false negatives in green): (a) undersegmentation error in Figure 2(a), (b) oversegmentation error in Figure 2(b), (c) undersegmentation and oversegmentation errors in Figure 2(c).

Most of interactive segmentation methods require the user's assistance to provide markers enabling to guide the segmentation process, from a parametric and/or geometric point of view, or to "correct" a rough segmentation initially performed in a manual fashion.

In this article, we focus on this second kind of issue, especially in the case where the structures of interest to be segmented are the ones of (local or global) extremal intensities, *i.e.*, the darkest or the brightest ones, which is a frequent case in several applications and especially in medical imaging. In order to illustrate this issue, let us consider the toy-example in Figures 1–3. The binary image depicted in Figure 1(b) is, for instance, the expected segmentation of the grey-level image depicted in Figure 1(a) for a given set of semantic elements. A user, when performing an initial (manual) segmentation of the same image, in an inaccurate but fast and easy-to-do fashion, will generally obtain results such as the ones proposed in Figure 2 which roughly approximate the desired result. Based on the user-dependent way this initial segmentation is performed, we may obtain different kinds of initial segmentations: undersegmentation (Figure 2(a)), oversegmentation (Figure 2(b)), or mixed under/oversegmentation (Figure 2(c)). Such results will lead to the generation of false positives and/or false negatives (see Figure 3). A crucial issue then consists in removing, automatically, or at least quite easily in an interactive way, these errors in order to finally obtain a correct segmentation result.

In this article, we propose a method enabling to tackle this problem by considering the component-tree of the image to be segmented, and by *finding the best segmentation induced by this component-tree with respect to an initial approximate segmentation provided by the user*. More formally, we propose to answer the following question:

(Q) Let I be a grey-level image (Figure 1(a)), let T be its component-tree, and let M be a binary object defined on

the same domain as I (a manual segmentation, Figure 2); how can we determine a part of T, and thus a part S of I (Figure 1(b)) which enables to fit at best M with the lowest computational cost?

This "best" segmentation can, in particular, be considered from a quantitative point of view, by finding a solution S which minimises the amount of false positives/negatives between S and M, which is actually supposed to be close to the correct result.

Such a question can be crucial, for instance to develop image analysis procedures involving ground-truth data [13] or to propose assisted-segmentation procedures such as the ones which will be described in the last sections.

The sequel of this article is organised as follows. Section 2 presents related works dealing with component-trees in image analysis and with interactive segmentation methods. In Section 3, usual definitions related to digital imaging and component-trees are recalled, making this article self-contained. The contributions of this article are located in Sections 4–6:

- A theoretical study (Section 4) is proposed to answer question (Q), and to prove the algorithmic efficiency of the proposed solution.
- An interactive segmentation methodology, based on these theoretical results is described (Section 5).
- The methodology is applied on medical images, namely cerebral magnetic resonance data of adults and foetuses (Section 6). Its efficiency is then assessed by comparison to other strategies.

Concluding remarks and perspectives will be found in Section 7. For the sake of readability and concision, all the proofs are provided in Appendix A, at the end of the article.

2. Related works

This section presents previous works devoted to the use of component-trees for image processing/analysis and an overview of interactive segmentation methods, then enabling to position the proposed work in its methodological and applicative context.

2.1. Component-trees

Component-trees have been involved in the development of several applications related to image processing and analysis. Most of these methods are devoted to filtering and/or segmentation[1]. In particular, a large part of them have been developed for (bio)medical image processing: vessels [19, 20, 21] and brain structures [22] segmentation from 3D magnetic resonance or X-ray data, melanocytic nevi segmentation from dermatological photographs [23], neuron filtering in 3D confocal microscopy or extraction of protein chains from 2D data [24]. Other methods have also been developed for a large spectrum of application fields: segmentation of video data [4], segmentation of wood micrograph [25], segmentation of astronomical data [17] and document analysis [14].

Note that other kinds of image processing applications have been considered, in particular, image registration [26, 3], image compression [4], image retrieval [27, 28], image classification [29], interactive 3D visualisation [30], multithresholding [16] or document binarisation [31].

In the field of filtering/segmentation, all the proposed methods have been designed to detect the structures of interest by using information related to the value of attributes stored at each node of the tree. In such strategies, an attribute or, more generally, a set of attributes, are chosen according to the hypotheses related to the applicative context. These attributes are assumed to model some characteristic properties of the structures of interest, and can be used in different ways:

- the desired values of the attributes (or the number of nodes of maximal value [22]) can be chosen by the user in order to select the relevant nodes and thus obtain the associated segmentation [19, 20, 17, 24];

[1] Note that the distinction between filtering and segmentation based on component-trees is generally not clearly defined, since segmentation is performed in a filtering fashion (see Section 4.1 for more details).

- the relevant values can be determined by analysing the signature of the attribute, *i.e.*, the evolution of their value with respect to the grey-level of the nodes [25];
- the relevant values can be learnt from examples, *e.g.*, by providing segmented ground-truth directly characterising the shape of the objects to be detected [23], or by feeding a classification process, in particular when the set of attributes becomes to large [29, 13, 14].

In these works, component-trees have then only been used for their ability to discriminate nodes with respect to attributes, thus leading to automated/parametric segmentation methods.

It is however possible to directly use the component-tree structure by taking advantage of the spatial/photometric decomposition of the image into nodes/connected components that it provides, in order to perform interactive segmentation. Methods based on such an alternative strategy should compute a segmentation result, no longer thanks to node attributes, but to a user-defined approximate result, which should then be matched at best by a relevant set of nodes, as discussed in Section 1. To the best of our knowledge, the methodology proposed in the next sections is the first one based on this strategy.

2.2. Interactive segmentation

Automatic segmentation algorithms do not always provide accurate results and are sometimes not sufficiently robust, especially in application fields in which images are prone to variability. Moreover, it is often crucial to achieve a precise segmentation of an image (for instance, to build ground-truths or in the context of computer-aided diagnosis) in a reasonable time.

The design of interactive segmentation methods is then an active research field and provides a solution to this by allowing a user to guide and/or refine the segmentation in an interactive fashion. The purpose of an interactive segmentation algorithm is to speed up the time consuming process of manual segmentations (which are, moreover, prone to errors and imprecision in the localisation of object contours). Interactive segmentation algorithms can also help to ensure the reproducibility between different segmentations of identical images, provided for example by different experts. Several ways to introduce interaction in a segmentation process have been considered, depending on the segmentation strategy:

- model initialisation, fine tuning of parameters and eventually interactive model refinement in the context of deformable or level-set based models [32, 33];
- delineation of the object of interest by successive interactive selections of optimal boundaries between the current cursor position and previously specified seeds positions in the context of intelligent scissors or live-wire algorithms, which are based on an optimal path search in a weighted graph [34, 35];
- manual depiction of foreground and background markers in the context of watershed algorithm [36, 37], seeded-region growing based algorithms [38, 39], graph-cuts [40, 41] or binary-partition tree algorithms [42, 43]. A comparative evaluation of interactive segmentation algorithms based on interactive marker selection is available in [44].

It could also be noted that most segmentation strategies can be used either in an unsupervised or semi-supervised fashion. By contrast with the above methods, the algorithm proposed in this paper involves only the rough delineation of foreground objects (and not background parts) and is based on a single scalar parameter (α) allowing to quickly browse among the different possible segmentations. Moreover, we focus in this article on the frequent case where the structures of interest to be segmented are the ones of locally extremal intensities.

3. Background notions

In this section, we recall standard notions related to component-trees. For complementary details, the reader may also refer to [4, 25, 15].

3.1. Definitions and notations

Let $n \in \mathbb{N}^*$. Let us consider an adjacency relation on the discrete grid defined by $\mathbb{Z}^n$, for instance, the $2n$- or the $(3^n - 1)$-adjacency, *i.e.*, the 4- (resp. the 6-) or the 8- (resp. the 26-) adjacency in $\mathbb{Z}^2$ (resp. $\mathbb{Z}^3$). Let $X \subseteq \mathbb{Z}^n$ be a non-empty set of $\mathbb{Z}^n$.

We say that two points $x, y \in X$ are connected (in X), and we note $x \sim_X y$, if there exists a sequence $(x_k)_{k=1}^t$ ($t \geq 1$) of elements of X such that $x_1 = x$, $x_t = y$ and x_k, x_{k+1} are adjacent for all $k \in [\![1, t-1]\!]$. Note that $\sim_X$ is an equivalence relation on X. The connected components of X are the elements of the quotient set $X/\sim_X$ (noted $\mathcal{C}[X]$ in the sequel). We say that X is connected if $\mathcal{C}[X] = \{X\}$.

Let $E \subset \mathbb{Z}^n$ be a finite connected set. Let $\bot < \top \in \mathbb{Z}$. Let $V = [\![\bot, \top]\!] \subset \mathbb{Z}$. A discrete grey-level image I can be defined as a function $I : E \to V$ (we also note $I \in V^E$).

For any $v \in V$, we define the thresholding function X_v by

$$\left| \begin{array}{ccccc} X_v & : & V^E & \to & \mathcal{P}(E) \\ & & I & \mapsto & \{x \in E \mid v \leq I(x)\} \end{array} \right.$$

where $\mathcal{P}(E) = \{Y \mid Y \subseteq E\}$.

For any $v \in V$, and any $X \subseteq E$, we define the cylinder function $C_{X,v}$ by

$$\left| \begin{array}{ccccc} C_{X,v} & : & E & \to & V \\ & & x & \mapsto & \left\{ \begin{array}{ll} v & \text{if } x \in X \\ \bot & \text{otherwise} \end{array} \right. \end{array} \right.$$

A discrete image $I \in V^E$ can then be expressed as

$$I = \bigvee_{v \in V} C_{X_v(I),v} = \bigvee_{v \in V} \bigvee_{X \in \mathcal{C}[X_v(I)]} C_{X,v}$$

where $\bigvee$ is the pointwise supremum for the sets of functions.

3.2. Component-trees

Let $\mathcal{K} = \bigcup_{v \in V} \mathcal{C}[X_v(I)]$ be the set of all the connected components obtained from the different thresholdings of I at values $v \in V$. The inclusion relation $\subseteq$ is then a partial order on $\mathcal{K}$. Let $v_1 \leq v_2 \in V$. Let $B_1, B_2 \subseteq E$ be the binary images defined by $B_k = X_{v_k}(I)$ for $k \in \{1, 2\}$. Let $C_2 \in \mathcal{C}[B_2]$ be a connected component of B_2. Then, there exists a (unique) connected component $C_1 \in \mathcal{C}[B_1]$ of B_1 such that $C_2 \subseteq C_1$.

Based on these properties, it can be easily deduced that the Hasse diagram of the partially ordered set $(\mathcal{K}, \subseteq)$ is a tree (*i.e.*, a connected acyclic graph), and more especially a rooted tree, the root of which is the supremum $X_\bot(I) = E$. This tree is called the *component-tree* of I.

Definition 1. *Let $I \in V^E$ be a grey-level image. The* component-tree *of I is the rooted tree $T = (\mathcal{K}, L, R)$ such that:*

$$\mathcal{K} = \bigcup_{v \in V} \mathcal{C}[X_v(I)]$$

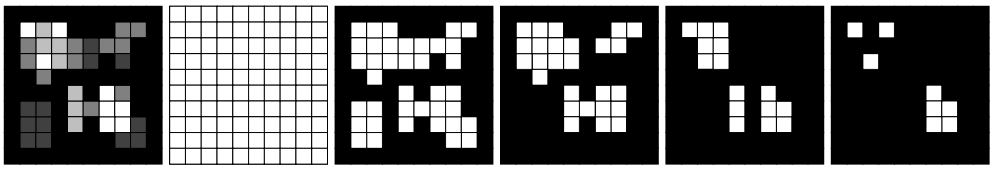
$$L = \left\{ (X, Y) \in \mathcal{K}^2 \left| \begin{array}{l} Y \subset X \wedge \forall Z \in \mathcal{K}, \\ Y \subseteq Z \subset X \Rightarrow Y = Z \end{array} \right. \right\}$$

$$R = \sup(\mathcal{K}, \subseteq) = X_\bot(I) = E$$

The elements of $\mathcal{K}$ (resp. of L) are the nodes *(resp. the* edges*) of T. The element R is the* root *of T. For any $N \in \mathcal{K}$, we set*

$$ch(N) = \{N' \in \mathcal{K} \mid (N, N') \in L\}$$

which is the set of the children *of the node N in T.*



(a) (b) (c) (d) (e) (f)

Figure 4: (a) A grey-level image $I : [\![0,9]\!]^2 \to [\![0,4]\!]$ (from 0, in black, to 4, in white). (b–f) Threshold images $X_v(I)$ (white points) for v varying from 0 (b) to 4 (f).

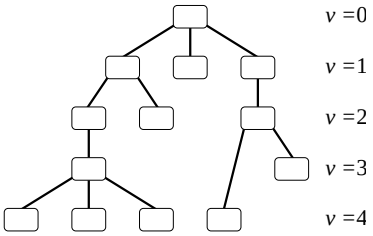


Figure 5: The component-tree of I (see Figure 4(a)). Its levels correspond to increasing thresholding values v. The root (*i.e.*, the upper node located at the level $v = 0$) corresponds to the support $E = [\![0,9]\!]^2$ of the image.

An example of component-tree defined for a 2D image is illustrated in Figures 4 and 5.

Each node of $\mathcal{K}$ is a binary connected component distinct from all the other nodes. However, such a connected component can be an element of $C[X_v(I)]$ for several (successive) values $v \in V$. For each $X \in \mathcal{K}$, we set

$$m(X) = \max\{v \in V \mid X \in C[X_v(I)]\} = \min_{x \in X}\{I(x)\}$$

We then consider that X is "associated" to the value $m(X)$, *i.e.*, to the highest value of V which generates this connected component. This choice is justified by reconstruction considerations which will be detailed in Section 4.1. An example of such an element X can be observed in Figures 4(e) and (f) (on the right side of the images), where it corresponds to a same binary connected component, while it generates only one node at the level $v = 4$ (which is the child of a node at the level $v = 2$) in the component-tree depicted in Figure 5.

The following definition, establishing "correlation scores" between a node and a given binary object, will be useful in the sequel of the article.

Definition 2. *Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I. Let $N \in \mathcal{K}$ be a node of T. Let $G \subseteq \mathbb{Z}^n$ be a binary object. We set*

$$\begin{aligned} p(N,G) &= |N \cap G| \\ p^*(N,G) &= |(N \setminus \bigcup_{N' \in ch(N)} N') \cap G| \\ n(N,G) &= |N \setminus G| \\ n^*(N,G) &= |(N \setminus \bigcup_{N' \in ch(N)} N') \setminus G| \end{aligned}$$

The value $p(N,G)$ (resp. $n(N,G)$) is the number of points of N which belong (resp. do not belong) to G. Note in particular that

$$p(N,G) + n(N,G) = |N|$$

The value $p^*(N,G)$ (resp. $n^*(N,G)$) is the number of points of N which belong (resp. do not belong) to G and which do not belong to any children of N. Note in particular that

$$p^*(N,G) + n^*(N,G) = |N \setminus \bigcup_{N' \in ch(N)} N'|$$

Remark 3. *When building the component-tree of I, it is possible to store, at each node $N \in \mathcal{K}$, the set of points*

$$E_N = N \setminus \bigcup_{N' \in ch(N)} N' \tag{1}$$

This leads, in particular, to an algorithmically useful partition $\{E_N\}_{N \in \mathcal{K}}$ of E. In such conditions, for a given binary object $G \subseteq \mathbb{Z}^n$, the computation of all the $p(N,G)$, $p^*(N,G)$, $n(N,G)$ and $n^*(N,G)$ ($N \in \mathcal{K}$) can obviously be performed in linear time $O(|E|)$. In the sequel, we will assume that $p(N,G)$, $p^*(N,G)$, $n(N,G)$ and $n^*(N,G)$ have been computed and are then available for every node $N \in \mathcal{K}$.

3.3. Image processing based on component-trees

Component-trees can be used to develop image processing/analysis procedures based on filtering or segmentation strategies [25]. Such procedures generally consist in determining a subset $\mathcal{K}' \subseteq \mathcal{K}$ among the nodes of the component-tree $T = (\mathcal{K}, L, R)$ of a considered image $I : E \to V$.

When performing filtering, the (grey-level) resulting image $I_f : E \to V$ induced by this set of nodes $\mathcal{K}'$ can be reconstructed as

$$I_f = \bigvee_{X \in \mathcal{K}'} C_{X,m(X)} \tag{2}$$

For instance, let us consider again the image I of Figure 1(a), also depicted in Figure 6(a). Let $T = (\mathcal{K}, L, R)$ be the component-tree of I, depicted in Figure 6(b), and let $\mathcal{K}' \subseteq \mathcal{K}$ be a subset of nodes of T, depicted in Figure 6(c). By applying Equation (2) on this set of nodes, we will then generate a filtered image I_f corresponding to the one depicted in Figure 7(a). Note that in this image, the grey-levels of the three nodes are preserved since each node $X \in \mathcal{K}'$ is associated to a cylinder function of value $m(X)$ (equal, for these three nodes, to 2, 3 and 4 respectively).

When performing segmentation, the (binary) resulting image $I_s \subseteq E$ is defined as the union of the nodes of $\mathcal{K}'$, *i.e.*, as

$$I_s = \bigcup_{X \in \mathcal{K}'} X$$

By applying this equation on the set of nodes $\mathcal{K}' \subseteq \mathcal{K}$ of Figure 6(c), we will generate a segmented image I_s corresponding to the one depicted in Figure 7(b).

In this last context, the determination of the nodes to preserve is a complex issue, which can be handled by considering attributes (*i.e.*, qualitative or quantitative information related to each node) to characterise the nodes of interest. An alternative solution is to search the set of nodes $\mathcal{K}' \subseteq \mathcal{K}$ which enables to generate a binary object being as similar as possible to a given binary target (*e.g.*, an approximate segmentation obtained from a manual process). In the sequel of the article, we focus on this specific issue, which can be formalised as an optimisation problem.

4. Theoretical study

4.1. Problem to solve

Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I. Let $M \subseteq E$ be a binary image. Let d be a distance on $\mathcal{P}(E)$. Question (Q), in Section 1, can then be reformulated as follows.

(Q') How can we compute a set of nodes $\mathcal{K}' \subseteq \mathcal{K}$ such that $d(\bigcup_{N \in \mathcal{K}'} N, M)$ is minimal, *i.e.*, such that the best binary object which can be built from $\mathcal{K}$ is as close as possible to M?

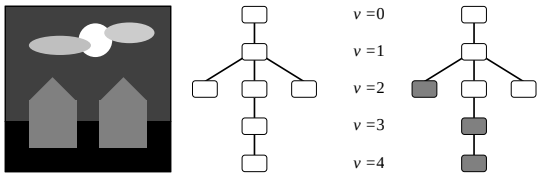


Figure 6: (a) A grey-level image. (b) The component-tree of $T = (\mathcal{K}, L, R)$ (a). (c) A set of nodes $\mathcal{K}' \subseteq \mathcal{K}$ selected from the component-tree T (in grey).

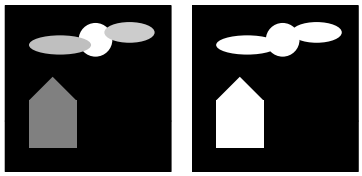


Figure 7: (a) Filtering result from Figure 6(a) based on the set of nodes $\mathcal{K}'$ depicted in Figure 6(c). (b) Segmentation result from Figure 6(a) based on the set of nodes $\mathcal{K}'$ depicted in Figure 6(c) (object in white).

More formally, the problem can be summarised as a minimisation problem, consisting in determining[2]

$$\widehat{\mathcal{K}} = \arg\min_{\mathcal{K}' \in \mathcal{P}(\mathcal{K})} \Big\{ d\Big(\bigcup_{N \in \mathcal{K}'} N, M \Big) \Big\}$$

An intuitive solution for determining a useful distance d is to consider the amount of false positives/negatives induced by $\bigcup_{N \in \mathcal{K}'} N$ with respect to the considered binary object of interest M.

Definition 4. *Let $\alpha \in [0, 1]$. Let $d^\alpha : \mathcal{P}(E) \times \mathcal{P}(E) \to \mathbb{R}^+$ be the function defined by*

$$d^\alpha(X, Y) = \alpha . |X \setminus Y| + (1 - \alpha) . |Y \setminus X| \tag{3}$$

The pseudo-distance[3] d^α constitutes a good similarity criterion between binary objects. Note that

$$d^0(X, Y) = |Y \setminus X|$$
$$d^1(X, Y) = |X \setminus Y|$$

i.e., $d^0(X, Y)$ (resp. $d^1(X, Y)$) is the amount of false negatives (resp. false positives) in X with respect to Y.

In the next sections, we will consider this distance. It will be established that it leads to algorithmically efficient processes, and satisfactory applicative results.

4.2. Preliminary properties

The following property directly derives from the definitions of Section 3.2.

Property 5. *Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I. Let $N \in ch(E)$. Let $\mathcal{K}_N = \{N' \in \mathcal{K} \mid N' \subseteq N\}$. Let $I_{|N} \in V^N$ be the grey-level image corresponding to the restriction of I to the node N. The Hasse diagram $(\mathcal{K}_N, L_N)$ of the partially ordered set $(\mathcal{K}_N, \subseteq)$ enables to define the component-tree $T_N = (\mathcal{K}_N, L_N, N)$ of $I_{|N}$ which is actually a subtree of T. Note in particular that $\{E\} \cup \{\mathcal{K}_N\}_{N \in ch(E)}$ is a partition of $\mathcal{K}$, while $\{(E, N)\}_{N \in ch(E)} \cup \{L_N\}_{N \in ch(E)}$ is a partition of L.*

[2]The notation arg min, used here for the sake of clarity, is potentially inaccurate since several sets of nodes P may minimise a given distance d.

[3]The function d^α is actually not a distance since $d^\alpha(X, Y) = d^\alpha(Y, X)$ if and only if $\alpha = 1/2$, $d^\alpha(X, Y) = 0 \Leftrightarrow X = 0$ if and only if $\alpha \in \,]0, 1[$, and d^α does not satisfy, in general, the triangle inequality.

Definition 6. *Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I. Let $x \in E$. We set*

$$\mathcal{K}_x = \{N \in \mathcal{K} \mid x \in N\}$$

which is the subset of all the nodes of $\mathcal{K}$ which contain x.

Since $E \in \mathcal{K}$, the following property is obvious, while the next one derives from the structure of the component-tree.

Property 7. *Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I. Let $x \in E$. Then, $\mathcal{K}_x$ is non-empty.*

Property 8. *Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I. Let $x \in E$. Then, $(\mathcal{K}_x, \subseteq)$ is a completely ordered set.*

Definition 9. *Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I. We define the function*

$$\left| \begin{array}{ccccc} \mathcal{G} & : & \mathcal{P}(\mathcal{K}) & \rightarrow & \mathcal{P}(E) \\ & & \mathcal{K}' & \mapsto & \bigcup_{N \in \mathcal{K}'} N \end{array} \right.$$

We set

$$\mathcal{Q} = \mathcal{G}(\mathcal{P}(\mathcal{K})) = \{\mathcal{G}(\mathcal{K}')\}_{\mathcal{K}' \subseteq \mathcal{K}}$$

which is the set of all the binary objects which can be generated from the subsets of nodes of $\mathcal{K}$.

Although there exist $2^{|\mathcal{K}|}$ distinct subsets $\mathcal{K}'$ of $\mathcal{K}$, most of these subsets generate a same binary object of E, more formally, we have $|\mathcal{Q}| \leq |\mathcal{P}(\mathcal{K})|$ (and generally $|\mathcal{Q}| \ll |\mathcal{P}(\mathcal{K})|$).

Property 10. *Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I. Let $\mathcal{Q}$ be the set of the objects which can be generated from the subsets of nodes of $\mathcal{K}$. Let $Q \in \mathcal{Q}$. Then, we have*

$$C[Q] = \min_{\subseteq} \mathcal{G}^{-1}(Q)$$

Less formally, the set of the connected components of Q is actually a subset of nodes of $\mathcal{K}$ which is included in any other subset of nodes $\mathcal{K}'$ of $\mathcal{K}$ generating Q. Such sets $\mathcal{K}'$ are then redundant (they contain in particular some nodes which are included in other nodes, and then useless for the generation of Q).

4.3. Main properties

4.3.1. Smallest superset / largest subset

In this subsection, we first focus on a specific case of the considered issue, which consists in finding a subset of nodes of the component-tree of an image I such that the object generated by these nodes is *included in* (resp. *includes*) the binary target G and is the *largest* (resp. the *smallest*) one verifying this property. This problem is equivalent to consider a distance d which only takes into account the amount of false negatives (resp. false positives) with respect to G (the link between such a distance d and the distance d^α of Definition 4 will be discussed in the next subsection).

From an applicative point of view, solving this specific problem makes sense when the initial rough segmentation proposed by the user is larger than (or smaller than) the desired result, as illustrated, for instance, by Figures 1(a,b), 2(a,b), and 3(a,b), in Section 1.

The following property establishes that there exists a (unique) solution to this problem.

Property 11. *Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I. Let $\mathcal{Q}$ be the set of the objects which can be generated from the subsets of nodes of $\mathcal{K}$. Let $G \subseteq E$. Then there exist $G^+, G^- \in \mathcal{Q}$ such that*

$$G^+ = \min_{\subseteq}\{Q \in \mathcal{Q} \mid G \subseteq Q\}$$
$$G^- = \max_{\subseteq}\{Q \in \mathcal{Q} \mid Q \subseteq G\}$$

We define now two functions which enable to compute these solutions G^+ and G^- (Definition 12, Properties 13 and 14) and we show that they authorise a computation in linear time with respect to the size (*i.e.*, the number of nodes) of the component-tree of the considered image I or the size of the support E of this image (Property 15).

Definition 12. *Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I. Let $G \subseteq E$. Let $\mathcal{F}^+, \mathcal{F}^- \in \mathcal{P}(\mathcal{K})^{\mathcal{K}}$ be the functions recursively defined, for all $N \in \mathcal{K}$, by*

$$\mathcal{F}^+(N) = \begin{cases} \{N\} & \text{if } p^*(N,G) \neq 0 \\ \bigcup_{N' \in ch(N)} \mathcal{F}^+(N') & \text{if } p^*(N,G) = 0 \end{cases}$$

$$\mathcal{F}^-(N) = \begin{cases} \{N\} & \text{if } n(N,G) = 0 \\ \bigcup_{N' \in ch(N)} \mathcal{F}^-(N') & \text{if } n(N,G) \neq 0 \end{cases}$$

In particular, if $ch(N) = \emptyset$, we have $\bigcup_{N' \in ch(N)} \mathcal{F}^-(N') = \bigcup_{N' \in ch(N)} \mathcal{F}^+(N') = \emptyset$, which guarantees the termination of these recursive definitions.

Property 13. *Let $\sigma \in \{+, -\}$. Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I. Let $G \subseteq E$. Then we have*

$$C[G^\sigma] = \mathcal{F}^\sigma(E)$$

The following property immediately derives from Property 13.

Property 14. *Let $\sigma \in \{+, -\}$. Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I. Let $G \subseteq E$. Then we have*

$$G^\sigma = \bigcup_{N \in \mathcal{F}^\sigma(E)} N$$

Property 15. *Let $\sigma \in \{+, -\}$. Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I. Let $G \subseteq E$. Then $C[G^\sigma]$ (and thus G^σ) can be computed with a linear algorithmic complexity $O(\max\{|\mathcal{K}|, |E|\})$, with respect to the number of nodes of the tree or the size of the image.*

4.3.2. General case

We now focus on the general case of the problem stated in Section 4.1, which consists in finding a set of nodes $\widehat{\mathcal{K}}$ of the component-tree of an image I verifying Equation (3), for the distance d^α proposed in Definition 4. The purpose is then to find the best compromise (according to a chosen weight $\alpha \in [0, 1]$) between the amount of false positives and false negatives with respect to a binary target G.

Since the set $\mathcal{Q}$ of the objects which can be generated from the subsets of nodes of a component-tree is finite, there necessarily exists a solution to this problem. Hereafter, we show that such a solution (Definition 16) can be computed in linear time with respect to the size (*i.e.*, the number of nodes) of the component-tree of the considered image I or the size of the support E of this image (Properties 19 and 20).

Definition 16. *Let $\alpha \in [0, 1]$. Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I. Let $G \subseteq E$. Let $\prec \in \{<, \leq\}$. Let $\mathcal{F}^\alpha : \mathcal{K} \to \mathcal{P}(\mathcal{K})$ and $c^\alpha : \mathcal{K} \to \mathbb{R}^+$ be the functions recursively cross-defined, for all $N \in \mathcal{K}$, by*

$$\begin{cases} \mathcal{F}^\alpha(N) = \{N\} \\ c^\alpha(N) = \alpha.n(N,G) \end{cases}$$

if $\alpha.n(N,G) \prec (1-\alpha).p^*(N,G) + \sum_{N' \in ch(N)} c^\alpha(N')$ and

$$\begin{cases} \mathcal{F}^\alpha(N) = \bigcup_{N' \in ch(N)} \mathcal{F}^\alpha(N') \\ c^\alpha(N) = (1-\alpha).p^*(N,G) + \sum_{N' \in ch(N)} c^\alpha(N') \end{cases}$$

otherwise.
In particular, if $ch(N) = \emptyset$, we have $\bigcup_{N' \in ch(N)} \mathcal{F}^\alpha(N') = \emptyset$ (which guarantees the termination of these recursive definitions), and $\sum_{N' \in ch(N)} c^\alpha(N') = 0$.

Definition 17. *Let $\alpha \in [0,1]$. Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I. Let $\mathcal{Q}$ be the set of the objects which can be generated from the subsets of nodes of $\mathcal{K}$. Let $G \subseteq E$. We define $G^\alpha \in \mathcal{Q}$ as*

$$G^\alpha = \bigcup_{N \in \mathcal{F}^\alpha(E)} N$$

From a reasoning similar to (and actually simpler than) the one of Property 13, we have the following result.

Property 18. *Let $\alpha \in [0,1]$. Let $I \in V^E$ be a grey-level image. Let $G \subseteq E$. Then we have*

$$\mathcal{F}^\alpha(E) = C[G^\alpha]$$

Property 19. *Let $\alpha \in [0,1]$. Let $I \in V^E$ be a grey-level image. Let $T = (\mathcal{K}, L, R)$ be the component-tree of I. Let $\mathcal{Q}$ be the set of the objects which can be generated from the subsets of nodes of $\mathcal{K}$. Let $G \subseteq E$. Then, we have*

$$d^\alpha(G^\alpha, G) = c^\alpha(E) = \min_{Q \in \mathcal{Q}} \{d^\alpha(Q, G)\}$$

Property 20. *Let $\alpha \in [0,1]$. Let $I \in V^E$ be a grey-level image. Let $G \subseteq E$. Then $\mathcal{F}^\alpha(E) = C[G^\alpha]$ (and thus G^α) can be computed with an algorithmic complexity $O(\max\{|\mathcal{K}|, |E|\})$, linear with respect to the number of nodes of the tree or the size of the image.*

Remark 21. *The set of nodes $\mathcal{F}^\alpha(E)$ and its associated binary object G^α enable to minimise $d^\alpha(., G)$, and thus to obtain an optimal solution to the issue considered in this work. However, $\mathcal{F}^\alpha(E)$ and G^α are generally not unique. To illustrate this assertion, let us consider the trivial case where $G = \emptyset$ (resp. $G = E$) and $\alpha = 0$ (resp. $\alpha = 1$). Obviously, in such a case, any set of nodes and any associated binary object minimise $d^0(., G)$ (resp. $d^1(., G)$), which is always equal to 0. However, the way to define $\prec$ in Definition 16 enables to break this non-determinism by choosing to favour the smallest ($<$) or the largest ($\leq$) solution (with respect to the inclusion relation $\subseteq$) among all the possible ones. In particular, if $\prec$ is set to $<$ (resp. to $\leq$) we have $\mathcal{F}^+ = \mathcal{F}^0$ (resp. $\mathcal{F}^- = \mathcal{F}^1$) (the easy proof of this assertion is left to the reader), which enables to establish a link between the studies of Sections 4.3.1 and 4.3.2.*

5. Methodology / technical details

5.1. Algorithmics

From the above study, which provides an answer to the question stated in Section 1, we can derive the method described in Algorithm 1. For the sake of simplicity, Algorithm 1 is described in a non-optimal but easy-to-understand way. In particular, the method is presented in an iterative fashion, while it is intrinsically recursive.

In its general form, the method corresponds to Definition 16, which solves the general case considered in Section 4.3.2. In the specific case where $\alpha = 0$ and $\prec = <$ (resp. $\alpha = 1$ and $\prec = \leq$), the method corresponds to the computation of $\mathcal{F}^+$ (resp. $\mathcal{F}^-$) in Definition 12, which solves the specific case of the smallest result including (resp. the largest result included in) the rough segmentation, considered in Section 4.3.1.

From this segmentation method, we can straightforwardly derive the (naive) interactive segmentation method described in Algorithm 2. By definition, Step 1 of this method presents a complexity $O(k.\max\{|\mathcal{K}|, |E|\})$, since Algorithm 1 has to be performed k times. Once this precomputation performed, the interactive choice of the result can be done by the user by inspection of the k proposed binary results.

By opposition to the case of a thresholding operation on a grey-level image, where $X_{v_2}(I) \subseteq X_{v_1}(I)$ whenever $\bot \leq v_1 < v_2 \leq \top$, we may think that sometimes $G^{\alpha_2} \not\subseteq G^{\alpha_1}$ while $0 \leq \alpha_1 < \alpha_2 \leq \top$. Fortunately, as stated by the following property, the increasing property of thresholding is actually inherited by the developed method.

Property 22. *Let $I \in V^E$ be a grey-level image. Let $G \subseteq E$. Let $\alpha_1 < \alpha_2 \in [0,1]$. Then we have $G^{\alpha_2} \subseteq G^{\alpha_1}$.*

Algorithm 1 - Segmentation method

Input
$I : E \to V$ (image to be segmented)
$R \subseteq E$ (rough segmentation of I)
$\alpha \in [0, 1]$ (weight parameter for false positives/negatives)
$\prec \in \{<, \leq\}$ (order involved in the cost minimisation formula)

Output
$G^\alpha \subseteq E$ (final segmentation of I)

Algorithm
1 - *Component-tree computation*
$T = (\mathcal{K}, L, R)$ (component-tree of I)
for all $N \in \mathcal{K}$ **do**
  $E_N = N \setminus \bigcup_{N' \in ch(N)} N'$
  $p^*(N, G) = |E_N \cap G|$
  $n(N, G) = |N \setminus G|$
end for
2 - *Cost minimisation*
for $v = \top$ to $\bot$ **do**
  for all $N \in \mathcal{K}$ such that $m(N) = v$ **do**
    if $ch(N) = \emptyset$ (*i.e.,* if N is a leaf) **then**
      if $\alpha.n(N, G) \prec (1 - \alpha).p^*(N, G)$ **then**
        $c^\alpha(N) = \alpha.n(N, G)$
        $\mathcal{F}^\alpha(N) = \{N\}$
      else
        $c^\alpha(N) = (1 - \alpha).p^*(N, G)$
        $\mathcal{F}^\alpha(N) = \emptyset$
      end if
    else
      if $\alpha.n(N, G) \prec (1 - \alpha).p^*(N, G) + \sum_{N' \in ch(N)} c^\alpha(N')$ **then**
        $c^\alpha(N) = \alpha.n(N, G)$
        $\mathcal{F}^\alpha(N) = \{N\}$
      else
        $c^\alpha(N) = (1 - \alpha).p^*(N, G) + \sum_{N' \in ch(N)} c^\alpha(N')$
        $\mathcal{F}^\alpha(N) = \bigcup_{N' \in ch(N)} \mathcal{F}^\alpha(N')$
      end if
    end if
  end for
end for
3 - *Result computation*
$G^\alpha = \bigcup_{N \in \mathcal{F}^\alpha(E)} N$

Algorithm 2 - Interactive segmentation method (naive version)

Input
$I : E \to V$ (image to be segmented)
$R \subseteq E$ (rough segmentation of I)
$\{\alpha_i\}_{i=1}^k$ (with $k \geq 2$) weight parameters (increasing values with respect to k)
$\prec \in \{<, \leq\}$ (order involved in the cost minimisation formula)

Output
$G^\star \subseteq E$ (final segmentation of I)

Algorithm
1 - *Segmentation results computation*
for all $i \in [\![1, k]\!]$ **do**
  Compute G^{α_i} by applying Alg. 1
end for
2 - *Segmentation choice*
Choose $G^\star \in \{G^{\alpha_i}\}_{i=1}^k$

Algorithm 3 - Interactive segmentation methods (choose either Step 2-a: "Pruning strategy" or Step 2-b: "Splitting strategy")

Input
$I : E \to V$ (image to be segmented)
$R \subseteq E$ (rough segmentation of I)
$\{\alpha_i\}_{i=1}^{k}$ (with $k \geq 2$) weight parameters (increasing values with respect to k)
$\prec \in \{<, \leq\}$ (order involved in the cost minimisation formula)

Output
$G^\star \subseteq E$ (final segmentation of I)

Algorithm
1 - *Component-tree computation*
See Step 1 of Alg. 1
2-a - *Segmentation results computation ("Pruning strategy")*
Compute $\mathcal{F}^{\alpha_k}$ and S^{α_k} from T by applying Steps 2 and 3 of Alg. 1
for i from $k-1$ to 1 **do**
  Prune T by removing from $\mathcal{K}$ all the successive children of $\mathcal{F}^{\alpha_{i+1}}$
  Compute $\mathcal{F}^{\alpha_i}$ and S^{α_i} from T by applying Steps 2 and 3 of Alg. 1
end for
2-b - *Segmentation results computation ("Splitting strategy")*
Compute $\mathcal{F}^{\alpha_1}$ and S^{α_1} from T by applying Steps 2 and 3 of Alg. 1
for i from 2 to k **do**
  $\mathcal{F}^{\alpha_i} = \emptyset$
  for all $N \in \mathcal{F}^{\alpha_{i-1}}$ **do**
    Let T_N be the subtree of T induced by N (see Property 5)
    Compute $\mathcal{F}_N^{\alpha_i}$ from T_N by applying Step 2 of Alg. 1
    $\mathcal{F}^{\alpha_i} = \mathcal{F}^{\alpha_i} \cup \mathcal{F}_N^{\alpha_i}$
  end for
  Compute G^{α_i} from $\mathcal{F}^{\alpha_i}$ (Step 3 of Alg. 1)
end for
3 - *Segmentation choice*
Let $S_k = \bigvee_{i=1}^{k} C_{S_{\alpha_i},i}$ (see Eq. 4)
Choose $G^\star \in \{G^{\alpha_i}\}_{i=1}^{k}$ by a standard thresholding of S_k

The first consequence of this property is the ability to store the k different results obtained for k increasing values $0 \leq \alpha_1 < \alpha_2 < \ldots < \alpha_{k-1} < \alpha_k \leq 1$, induced by a method such as the one described in Algorithm 2, as a grey-level image $S_k : E \to [\![1, k]\!]$ defined –similarly to the filtering process proposed in Section 4.1 (Equation 2)– by

$$S_k = \bigvee_{i=1}^{k} C_{G^{\alpha_i},i} \tag{4}$$

where $G^{\alpha_i} \subseteq E$ is the binary result of the segmentation method (Algorithm 1) for the parameter α_i. In such a situation, we can avoid to store k distinct binary images, and the interactive choice of the result by the user can be performed (of course, in real-time) by actually performing a standard thresholding of S_k among the values $[\![1, k]\!]$ (these values being associated to the set of parameters $\{\alpha_i\}_{i=1}^{k}$ by the trivial one-to-one mapping induced by Equation 4).

The second consequence of Property 22 is the possibility to optimise the computation of the k solutions G^{α_i} by taking advantage of the fact that $G^{\alpha_2} \subseteq G^{\alpha_1}$ whenever $\alpha_1 < \alpha_2$. Indeed, let us suppose that for $\alpha \in [0, 1]$, a given node $N \in \mathcal{K}$ belongs to set of nodes $\mathcal{F}^\alpha$ which generates the solution G^α, *i.e.*, that N is a connected component of G^α. Then, for any $\alpha^- < \alpha$ (resp. $\alpha^+ > \alpha$), N will necessarily be included in a connected component of G^{α^-} (resp. N will include all the connected components of G^{α^+} that it intersects). This implies in particular that two strategies can be considered for computing k solutions G^{α_i} with $0 \leq \alpha_1 < \alpha_2 < \ldots < \alpha_{k-1} < \alpha_k \leq 1$:

1. "Pruning strategy", where the sets $\mathcal{F}^{\alpha_i}$ are computed from $i = k$ to 1. Once $\mathcal{F}^{\alpha_i}$ has been computed, all the nodes which are children of a node of $\mathcal{F}^{\alpha_i}$ are removed from the current component-tree (since these nodes are no longer eligible for lower values of α). The next set of nodes $\mathcal{F}^{\alpha_{i-1}}$ is then computed from this pruned component-tree.

2. "Splitting strategy", where the sets $\mathcal{F}^{\alpha_i}$ are computed from $i = 1$ to k. Once $\mathcal{F}^{\alpha_i}$ has been computed, the component-tree associated to each sub-image induced by a connected component of G^{α_i} (*i.e.*, each node of $\mathcal{F}^{\alpha_i}$, see Property 5) is built. The next set of nodes $\mathcal{F}^{\alpha_{i+1}}$ is then computed as the union of the results of the process on each one of these component-trees (since the nodes which have no node of $\mathcal{F}^{\alpha_i}$ among their successive children are no longer eligible for higher values of α).

In the worst case (*i.e.*, when $G^{\alpha_k} = E$ for the "Splitting strategy" and G^{α_1} is only composed of leaves for the "Pruning strategy"), the algorithmic complexity of these strategies remains the one of Step 1 of Algorithm 2, namely $O(k.\max\{|\mathcal{K}|, |E|\})$. However, in real applications, we may reasonably suppose that the complexity will significantly decrease since the successive G^{α} will progressively become larger/smaller (enabling to reduce the size of the image to be processed and/or the number of considered nodes during the successive solution computations).

The two optimised versions of the interactive segmentation method are described in Algorithm(s) 3.

5.2. Software

This section describes the actual image segmentation tool which may be developed based on the theoretical and methodological studies presented above[4] .

5.2.1. Nodes selection

Component-tree computation. The component-tree is computed using Salembier's algorithm [4], which is based on a recursive flooding of an image from its maxima[5].

For each node N are stored two attributes related to the current marker G: $p^*(N, G)$ and $n(N, G)$. These attributes are computed during the tree computation, using a minor modification of Salembier's algorithm. For efficiency reasons and to avoid redundancies, each pixel is stored in only one node of the tree, which is the highest node containing the pixel. Therefore, for each node N is stored exactly the set of pixels defined by $E_N = N \setminus \bigcup_{N' \in ch(N)} N'$ The computation of $p^*(N, G) = |E_N \cap G|$ involves only pixels stored in node N: this attribute can therefore be updated each time a new pixel is stored in a node N during the component-tree computation. The computation of $n(N, G) = |N \setminus G|$ involves pixels which are not all stored in N. Therefore, each time a new pixel is stored in a node N is updated the value $n^*(N, G) = |E_N \setminus G|$ during the tree computation. A second pass is then necessary to compute $n(N, G)$ for each node, based on the property: $n(N, G) = \sum_{N' \in ch(N)} n^*(N', G)$. This is achieved using a depth-first scan of the tree nodes.

5.2.2. Interactive segmentation

Interactive segmentation is based on two information: the marker image and the value of the α parameter. Depending on the application, the user may interact on one or both of them.

In the case where only the α parameter is used, it can be advantageous to use the increasing property (Property 22) of the nodes selection procedure. This way, the results associated to various α sampled at regular values can be precomputed, in order to speed up the interactive and visualisation process. This method is summarised in Algorithm 3.

In the other case where both parameters (marker image and α value) are used, it can be more advantageous to recompute the segmentation result each time a new value of α is selected. In this case, the attributes $p^*(N, G)$ and $n(N, G)$ stored in each node need also to be recomputed each time a new marker image is validated. An interactive procedure can then be designed using the following scheme:

1. component-tree computation (automatic step);
2. manual drawing of a marker image (user interaction);
3. attributes computation (automatic step);
4. choice of an α value (user interaction) ;
5. result computation (automatic step);
6. if the result is not satisfactory, back to 2 (marker modification) or back to 4 (α modification with the same marker).

[4]Such a tool can be freely downloaded from the following url: `http://webloria.loria.fr/~naegelbe/index.php/software`

[5]Another efficient algorithm is Najman's one [15], which is based on Tarjan's union-find algorithm. This latter algorithm is particularly well suited for images having a large number of different values, but in our case it was slower than Salembier's algorithm on our validation images.

(a) Undersegmented marker (b) Mixed marker (c) Oversegmented marker

(d) $\alpha = 0.05$ (e) $\alpha = 0.12$ (f) $\alpha = 0.99$

Figure 8: (a,d), (b,e), (c,f) Examples of markers (in red) and associated segmentation for a given α (in blue). (Image is from the Berkeley Segmentation Dataset [45].)

Some examples of segmentation results obtained on a test image are depicted in Figure 8. They illustrate, in particular, the different kinds of markers which can be considered (the wide variability of markers being authorised by the α parameter).

6. Application to medical image analysis

The analysis of medical data (magnetic resonance imagery (MRI), computed tomography, etc.) is required for a large spectrum of applications, including for instance computer-aided diagnosis, patient follow-up, or presurgical planning. For such purposes, segmentation is generally a step of a complex pipeline involving both image processing and expert (*e.g.*, radiologist, surgeon) handling. In particular, the reliability of the results provided by this first segmentation step dramatically influences the quality of the whole analysis protocol.

In this section, we consider a classical issue in medical image analysis, namely the segmentation of the brain (*i.e.*, both grey and white matter) or the whole intracranial volume (*i.e.*, grey and white matter, plus the cerebrospinal fluid) from MRI data. This choice is justified by the following two arguments. First, segmentation algorithms applied to brain MRI can be accurately assessed by using simulated images from the commonly used BrainWeb database [46], which provides MRI images with their associated anatomical ground-truth. Such assessment are proposed in Section 6.1. Second, although the application of a segmentation method on simulated data constitutes a necessary prerequisite for its validation, its evaluation on real data remains fundamental. As the segmentation of the intracranial volume constitutes an important prerequisite, especially in foetal brain analysis from MRI data, it can then be interesting to evaluate the adequacy of the method to this task. This study is proposed in Section 6.2.

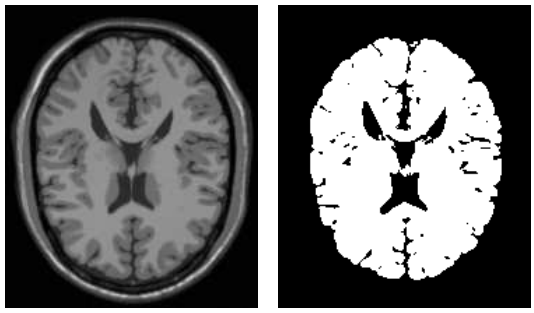
In both studies (on simulated and real images), the validation protocol will consider results obtained by the proposed method (denoted CT), and by the graph-cut method [40, 41] (denoted GC). We chose to restrict the comparison with existing "high-level" methods to the only graph-cut one, since (*i*) it has been often proved to be the most efficient among other interactive methods (in terms of computation time and result accuracy) [44], and (*ii*) it presents, for

the user, a *modus operandi* similar to the proposed component-tree method[6]. Note finally that, by opposition to the graph-cut method, the component-tree method is devoted to the case where the structures of interest to be segmented are the ones of (locally) extremal intensities. Such a case is however quite frequent in several application fields, and in particular in the ones proposed hereafter.

6.1. Simulated brain images

6.1.1. Images

BrainWeb[7] is a database, providing simulated normal brain realistic MRI data for several acquisition modalities (T1, T2, etc.) and acquisition parameters. Each image is provided with an anatomical ground-truth, which associates, in particular, each voxel of the intracranial volume to a specific tissue class, as illustrated in Figure 9(a,b).



(a) (b)

Figure 9: (a) A BrainWeb image (axial slice), and (b) its associated ground-truth (from light-grey to dark grey: white matter, grey-matter, cerebrospinal fluid, extracranial volume).

For the proposed experiments, the considered BrainWeb data have been chosen with classical acquisition parameters (with respect to a standard brain MRI acquisition), namely by considering T1-weighting, with $1 \times 1 \times 1$ mm resolution, 1 to 9% noise level, and 20 to 40% inhomogeneity field. Three (axial) slices have been chosen among the whole 3D image, at the top, middle and bottom of the brain, respectively. For each slice, six versions with different noise ratios and inhomogeneity fields have been considered. They are denoted S_i^n (where n and i corresponds to the noise and the intensity inhomogeneity levels, in %). These six versions (S_{20}^1, S_{20}^5, S_{20}^9, S_{40}^1, S_{40}^5, and S_{40}^9) are depicted in Figure 10, for the middle slice.

6.1.2. Segmentation protocol

The two considered segmentation methods have been applied as follows. For GC, the following two steps have been performed as many times as necessary: (*i*) manual drawing of both object and background markers, (*ii*) graph-cut processing[8]. For CT, the following two steps have been performed as many times as necessary: (*i*) manual drawing of object markers, (*ii*) node selection by tuning of α.

6.1.3. Computation time vs. *quality*

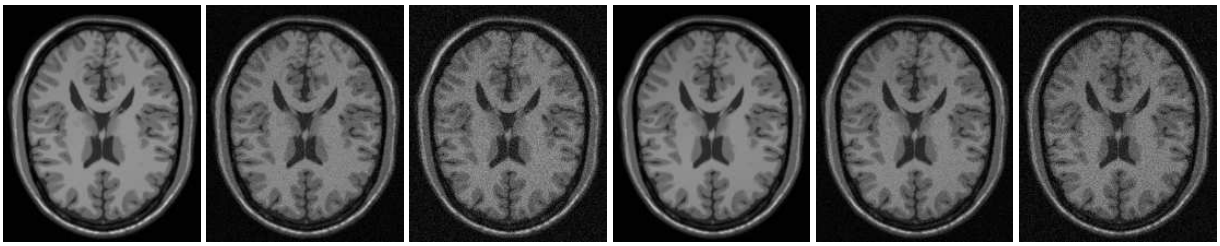
Experiments have been carried out in order to study the link between the quality of the segmentation results and the time required to obtain them.

Two measures have been used to evaluate the quality of the result:

[6]Note, however, that in the graph-cut method, the user has to mark two kinds of areas assumed to be included in the object and in the background, respectively, while in the component-tree method, the users has only to mark (and actually to roughly segment) the object.

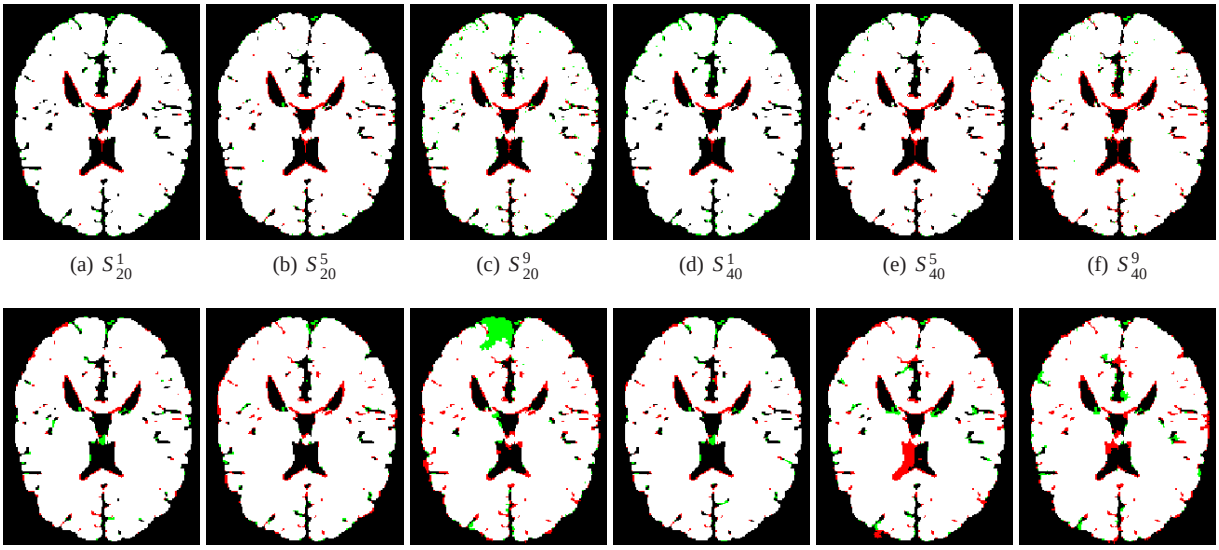
[7]`http://www.bic.mni.mcgill.ca/brainweb`

[8]The graph-cuts software considered here is the one proposed in [44] which requires to set five parameters. The values for these parameters, determined by preliminary tests on the considered images, have been: no smoothing, $\alpha = 1$, $\sigma = 1$, $\lambda = 0$, histogram quantisation = 1.



(a) S_{20}^1 (b) S_{20}^5 (c) S_{20}^9 (d) S_{40}^1 (e) S_{40}^5 (f) S_{40}^9

Figure 10: BrainWeb images S_i^n considered for the proposed validations (see text). The ground-truth proposed in Figure 9(b) is the one corresponding to this slice.



(a) S_{20}^1 (b) S_{20}^5 (c) S_{20}^9 (d) S_{40}^1 (e) S_{40}^5 (f) S_{40}^9

(g) S_{20}^1 (h) S_{20}^5 (i) S_{20}^9 (j) S_{40}^1 (k) S_{40}^5 (l) S_{40}^9

Figure 11: Segmentation results for Figure 10, with CT (a–f) and GC (g–l) after 120 seconds (zoomed images). In white: true positives. In red: false positives. In green: false negatives.

- the κ index defined by

$$\kappa = \frac{2.\mathrm{TP}}{2.\mathrm{TP} + \mathrm{FP} + \mathrm{FN}}$$

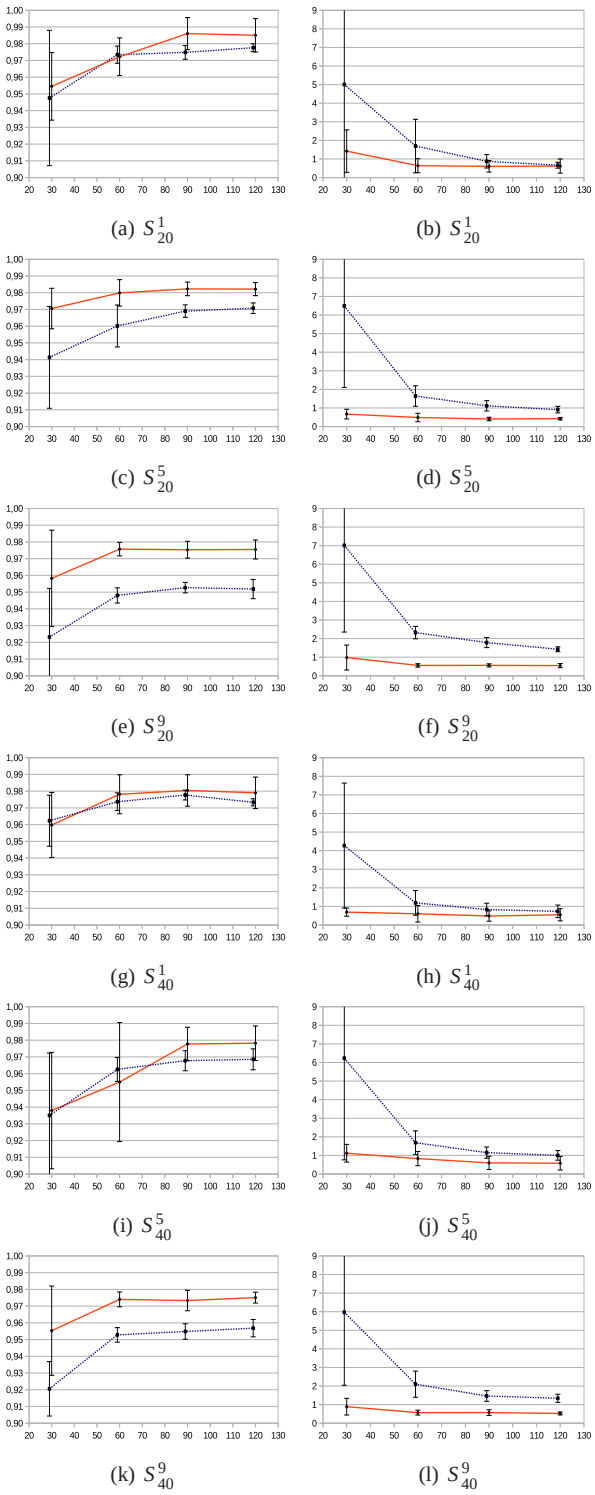
  where TP, FP and FN are respectively the amounts of true positives, false positives and false negatives with respect to the BrainWeb ground-truth image;

- the mean point-to-set distance D between the borders ∂S and ∂G of the segmentation result and the ground-truth, defined by

$$D = \frac{1}{|\partial S| + |\partial G|}\Big(\sum_{x \in \partial S} d(x, \partial G) + \sum_{x \in \partial G} d(x, \partial S) \Big)$$

  where d is a standard point-to-point distance (in our case, the Euclidean distance). This distance D is expressed in pixels in the sequel.

The evolution of the κ index and the mean point-to-set distance, with respect to time, obtained for each one of the six (n, i) configurations (by gathering the results of the three users on the three slices) are depicted in Figure 12.



(a) S^1_{20} (b) S^1_{20}

(c) S^5_{20} (d) S^5_{20}

(e) S^9_{20} (f) S^9_{20}

(g) S^1_{40} (h) S^1_{40}

(i) S^5_{40} (j) S^5_{40}

(k) S^9_{40} (l) S^9_{40}

Figure 12: Quality of the segmentation results on BrainWeb, depending on the time required to compute these segmentations, for CT (red full line) and GC (green dashed line). Each dot (resp. vertical line) is the mean value (resp. the mean standard deviation) of the results obtained by the different users. Left column: κ index. Right column: D.

Examples of segmentation results for the slice of Figure 10, obtained at the end of the process (120 seconds) are depicted in Figure 11 which emphasises in particular the false positives and negatives with respect to the BrainWeb ground-truth image.

6.2. Real foetal brain images

6.2.1. Applicative context

Most of knowledge concerning brain maturation is based on *post mortem* studies which do not allow joint analysis of anatomical structure development and cognitive development. The non-invasive nature of MRI provides unique opportunities for *in vivo* investigation of the developing human brain. In the case of foetuses, MRI is a valuable complement to prenatal sonography to confirm and characterise suspected brain abnormalities.

Image interpretation is generally performed based on foetal brain atlas book and neuro-paediatricians have to make the correspondence mentally between 3D MRI data and 2D printed histological images which is tedious and error prone.

The development of ultrafast 2D acquisition sequences has led to significant improvements in the clinical utility of foetal MRI [47]. However, the slice acquisition time is still very critical and has to be as short as possible to reduce the impact of foetal motion on the exam, since foetal MRI is often performed without sedation. As a result, sets of thick 2D slices are generally acquired in clinical studies and interpretation remains limited by visual inspection.

In the context of foetal MRI study, removal of non-brain tissues in MR images (also known as skull stripping [48, 49, 50]) is an important step in enabling accurate measurement of brain structures. While this is a crucial step for morphometry studies, it remains an open issue, especially for foetal MRI where the region of interest is surrounded by many other structures.

6.2.2. Images and ground-truth

The considered images are foetal MR scans: T2 weighted HASTE sequence (TE/TR = 147/3190 ms) on a 1.5 T Siemens Avanto MRI Scanner (SIEMENS, Erlangen, Germany), resolution : $0.74 \times 0.74 \times 3.45$ mm. An example of foetal brain MRI[9] is provided in Figure 13 (left column).

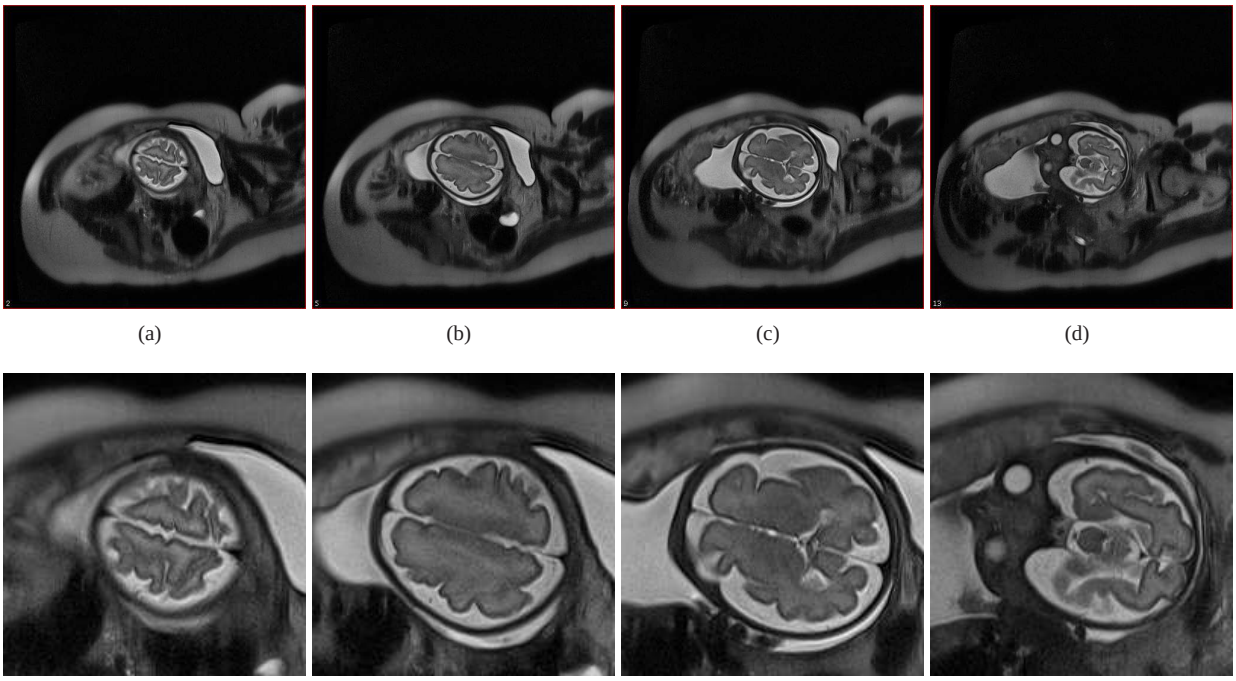
By opposition to the validations performed on simulated data in the previous subsection, there is –by definition– no ground-truth directly available here. The considered ground-truth is then the one provided by manual segmentation carried out by medical experts.

These six versions (S^1_{20}, S^5_{20}, S^9_{20}, S^1_{40}, S^5_{40}, and S^9_{40}) are depicted in Figure 10, for the middle slice.

6.2.3. Segmentation protocol

The CT and GC methods have been applied as in Section 6.1, still on 2D slices, while (by opposition to the above study) the whole intracranial volume has been processed for each considered image. Although foetal brain MR data are actually 3D ones, this 2D slice-by-slice approach is actually justified by the following two facts: (*i*) the size of foetal brains (approximately $100 \times 80 \times 70$mm for the considered data), and the resolution of the data (approximately 4mm interslice distance) generate only a small number of slices in the 3D volume (approximately, 15 slices), and (*ii*) the possible movements of the foetus during the acquisition process may result in spatial inaccuracies between successive slices (*e.g.*, translations, rotations), making the spatial continuity assumption (verified for adults brain images and justifying 3D segmentation approaches) irrelevant here. Moreover, note that, by opposition to Section 6.1, the considered results have been the ones consisting of the areas bounded by the external curve generated by the segmentation results of CT and GC (indeed, for skull stripping, the relevant information is the boundary of the intracranial volume).

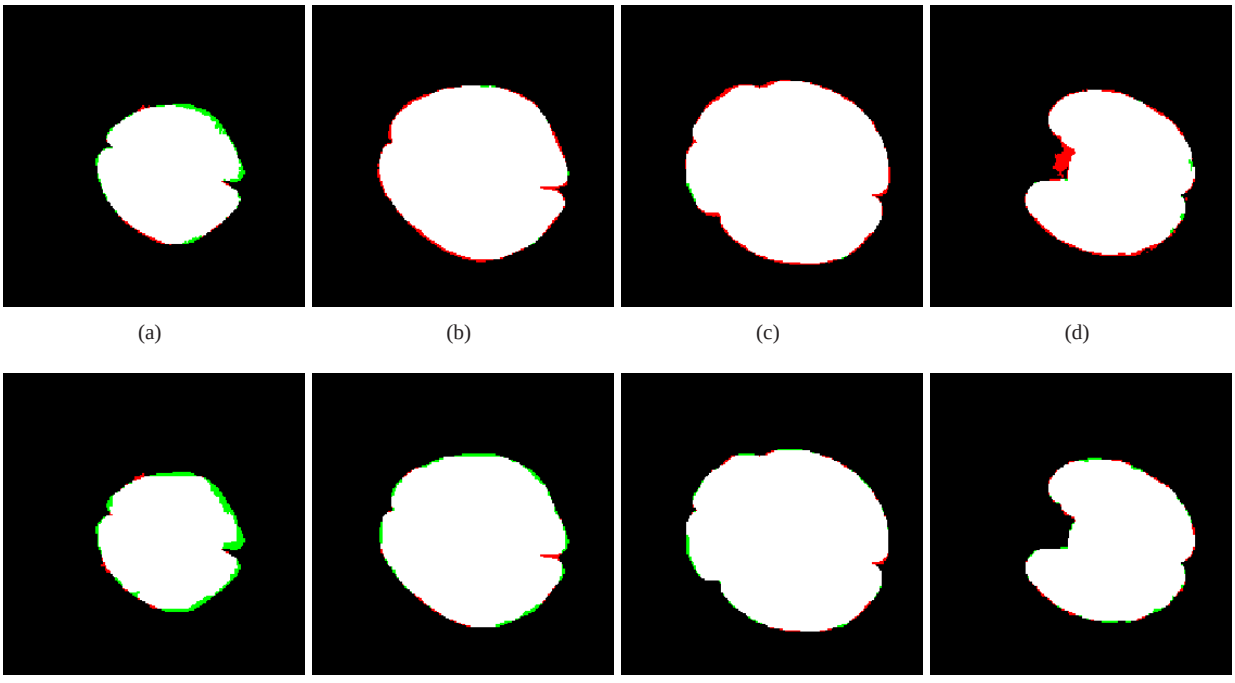
[9] Note that in Section 6.1 (see Figures 9 and 10), which deals with adult brains, the white matter, grey-matter and cerebrospinal fluid appear in light grey, dark grey and black, respectively, while in the current case which corresponds to foetus brains, these tissues appear in light grey, dark grey and white, respectively.



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 13: (a–d) Foetus brain images (subset of axial slices sampled in the whole image). (e–h) Zoom on the cerebral part of (a–d).



(a) (b) (c) (d)

(e) (f) (g) (h)

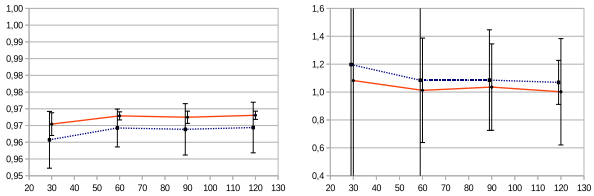
Figure 14: Segmentation results for Figure 13(e–h), with CT (a–d) and GC (e–h) after 60 seconds. In white: true positives. In red: false positives. In green: false negatives.

6.2.4. *Computation time* vs. *quality*

The same experiment as in Subsection 6.1.3 has been carried out in order to study the link between the quality of the segmentation results and the time required to obtain them. For each one of the 17 slices of the 3D image (partially visualised in Figure13 (left column)), several segmentations have been performed, by 3 different users, with both CT and GC.

The evolution of the κ index and the mean point-to-set distance, with respect to time (obtained by gathering the results of the three users on the 17 slices) are depicted in Figure 15.

Examples of segmentation results, obtained at the end of the process (60 seconds) are depicted in Figure 11 which emphasises in particular the false positives and negatives with respect to the BrainWeb ground-truth image.



(a) (b)

Figure 15: Quality of the segmentation results on foetus images, depending on the time required to compute these segmentations, for CT (red full line) and GC (green dashed line). Each dot (resp. vertical line) is the mean value (resp. the mean standard deviation) of the results obtained by the different users. (a) κ index. (b) D.

6.3. *Discussion*

From the results of these experiments, performed on both synthetic and real images, several remarks can be made.

For simulated and *in vivo* MR data, one may notice that the final results (obtained at 120 seconds for BrainWeb, and 60 seconds for foetuses) are generally very similar (see Figures 11 and 14), despite intrinsically different *modus operandi* (GC tries to fit at best the boundary of the object on highest gradient values of the image, thus following a "1D approach", while CT tries to fit at best the object markers in the image, thus following a "2D approach"). From a qualitative point of view, the (intermediate and final) results are nearly always better with CT than with GC (with an increase in quality for the considered criteria κ and D which is small but however proportionnaly significant with respect to the small gap between the obtained and optimal values). Note in particular that BrainWeb experiments tend to show that the quality improvement between CT and GC becomes higher when the noise ratio increases.

One may notice that CT is robust to noise (at least up to 9% in the current experiments), and to intensity inhomogeneity (at least up to 40% in the current experiments), while it does not integrate any regularisation scheme in order to correct such image artifacts. For the robustness of CT to noise, this can be explained by three facts. First, noise cannot generate false positive results out of the markers, since the process implies that a binary connected component must, at least, intersect (and thus, for noise, "be included in") the marker, in order to belong to the segmentation result. Second, noise of value higher (resp. lower) than the structures of interest of high (resp. low) values, included in the markers, generally generate binary connected components which are included in relevant connected components obtained by thresholding at a lower (resp. higher) value. Then, except in cases where the marker is a quite large superset of the searched structure (see Figure 8(c,f) for an example), CT is not altered by such noise. Third, noise of value lower (resp. higher) than the structures of interest of high (resp. low) values, included in the markers, may possibly generate "holes" in relevant connected components obtained by thresholding at a high (resp. low) value. However, in order to generate such false negatives, it is necessary for such noise to have a relative value difference larger than the gap between the value of the structure of interest and the value of its neighbouring background, which is possible, but statistically unfrequent (see the small white points in Figure 8(d,e) for an example). Moreover, the robustness of CT to intensity inhomogeneity can also be justified by the fact that methods based on connected filtering become sensitive to such effects only when the intensity of the structures of interest and the intensity of their direct neighbouring background have a nonempty intersection, which is the case only for images where the contrast is quite low and/or where the intensity inhomogeneity becomes huge (which is generally not the case, even in medical imaging).

From a time cost point of view, CT and GC both converge rapidly and in comparable times (asymptotic results reached at 60s, with a low improvements –and sometimes a quality decrease– past this time). For very low values (here 30s), the results of CT are more satisfactory (better mean value), and more homogeneous between users (lower standard deviation) than those of GC. It may however be noticed that for larger 2D images, and *a fortiori* for 3D images, the computational cost of CT may become lower than the one of GC. Indeed, GC relies on classic techniques of maximal flow/minimal cut computation [51] (improved by optimised versions [52]) which lead to obtain a result in (low order) polynomial time [41] with respect to the size of the image. By comparison, as stated in Property 20, the proposed CT method can be run in linear time with respect to the size of the image (since the number of nodes in a component-tree is generally lower than the size of the image). In such conditions, the relevance to use a CT segmentation method increases when the size of the image becomes high.

Finally, since:

(*i*) CT presents computation times similar to GC for 2D images, and has a lower algorithmic complexity which may lead to better computation times for larger images;

(*ii*) it only requires to determine one family of markers ("objects") *vs.* two families of markers ("objects" and "background") for GC, which may in particular lead to involve CT in the developement of example-based segmentation strategies (for instance for the segmentation of 3D images);

(*iii*) it is parameter free (except α which has to be tuned during the segmentation process, and is thus not of same nature than the five predetermined GC parameters), by opposition to GC;

it can be concluded that CT is a relevant segmentation tool for images where the structures of interest correspond to photometric local maxima, even in cases of noise and intensity inhomogeneity. In this context, it presents an ergonomic alternative to the state-of-the-art GC.

7. Conclusion

In this article, an original methodological scheme, based on component-trees, has been proposed for segmentation purpose. By opposition to the other existing approaches based on component-trees, it does not rely on the use of knowledge modelled by attributes stored at each nodes of the tree (which enables to decide which ones have to be preserved to generate the segmentation result), but on a user-defined raw segmentation from which the most relevant nodes are extracted to obtained a refined result.

From a theoretical point of view, it has been proved that, in such a strategy, these relevant nodes could be discriminated in linear time (with respect to the size of the component-tree). Based on this result, efficient algorithms have been proposed, finally leading to an interactive segmentation method. This method has been assessed in the experimental context of both adult and foetal brain analysis from MRI slices. In the validity area of component-trees, namely in cases where structures of interest present locally maximal intensity values, these experiments have emphasised the robustness of the method in terms of segmentation accuracy, its fastness and its ergonomy, in particular by comparison to the state-of-the-art graph-cuts algorithm.

The method is currently designed to perform segmentation of nD images based on an initial nD raw segmentation (with any $n \geq 1$). It is then particularly well-fitted for processing 2D data (the raw segmentation of which is quite easy). In the context of 3D image segmentation, the proposed methodological scheme may naturally be involved for the design of example-based segmentation, where a 3D segmentation example (*e.g.*, registered atlas, raw segmentation obtained or another method, etc.) may replace the manual segmentation provided in a manual fashion in the 2D case. Consequently, further works will now consist of providing such 3D extensions of this method.

From a more theoretical point of view, research efforts will also be devoted to study the impact of using non-standard connectivity (such as second-generation connectivities [53], and in particular mask-based ones [24]) on the behaviour of the proposed method. Moreover, an extension of the proposed methodology to the case of the *level line trees* [54], which provide an auto-dual representation of the image structure into level-sets, and is then better suited for segmentation applications involving structures of interest which do not present locally extremal values in the considered images, may also be considered.

Acknowledgements

The research leading to these results has received funding from the European Research Council under the European Community's Seventh Framework Programme (FP7/2007-2013 Grant Agreement no. 207667), and from the French *Agence Nationale de la Recherche* (Grant Agreement ANR-2010-BLAN-0205). This work is also funded by NIH Grant R01 NS055064 and a CNRS grant for collaboration between LSIIT and BICG.

A. Appendix : Proofs

Proof of Property 11 If $G = \emptyset$, by setting $G^+ = \emptyset \in \mathcal{Q}$, we are done. Let us now suppose that $G \neq \emptyset$. For any $x \in G$, we set $N_x = \min_{\subseteq} \mathcal{K}_x$.

Let $G^+ = \bigcup_{x \in G} N_x$. Then we have $G^+ \in \mathcal{Q}$ and $G \subseteq G^+$. Let $Q' \in \mathcal{Q}$ such that $G \subseteq Q'$. Let $y \in G^+$. If $y \in G$, then we have $y \in Q'$. Let us now suppose that $y \in G^+ \setminus G$. Then, there exists $x \in G$ such that $y \in N_x$. Since $x \in G \subseteq Q'$, there exists $N \in \mathcal{K}_x$ such that $N \subseteq Q'$. But then, we have $y \in N_x \subseteq N \subseteq Q'$. Consequently, we have $Q \subseteq Q'$, and thus $G^+ = \min_{\subseteq}\{Q \in \mathcal{Q} \mid G \subseteq Q\}$.

Let $G^- = \bigcup_{N \in \mathcal{K} \wedge N \subseteq G} N$. We have $G^- \in \mathcal{Q}$ and $G^- \subseteq G$. Let $Q' \in \{Q \in \mathcal{Q} \mid Q \subseteq G\}$. Let us suppose that there exists $x \in Q' \setminus G^-$. In particular, we have $x \in G$. There exists $N_x \in \mathcal{K}_x$ such that $N_x \subseteq Q'$. If $N_x \subseteq G$ then we have $x \in N_x \subseteq G^-$: contradiction. If $N_x \not\subseteq G$ then we have $Q' \not\subseteq G$: contradiction. Consequently, for all $x \in Q'$, we have $x \in G^-$, and thus $G^- = \max_{\subseteq}\{Q \in \mathcal{Q} \mid Q \subseteq G\}$. □

Proof of Property 13 Let $X, Y \in \mathcal{F}^\sigma(E)$. By definition, we have $X, Y \in \mathcal{K}$. Moreover, if $X \neq Y$, it obviously comes that $X \cap Y = \emptyset$. Consequently, there exists $Q \in \mathcal{Q}$ such that $\mathcal{F}^\sigma(E) = C[Q]$. By induction from the definition of $\mathcal{F}^+(E)$ and $\mathcal{F}^-(E)$, we easily deduce that

$$\bigcup_{N \in \mathcal{F}^+(E)} N = \bigcup_{N \in \mathcal{K} \wedge p^*(N,G) \neq 0} N$$

$$\bigcup_{N \in \mathcal{F}^-(E)} N = \bigcup_{N \in \mathcal{K} \wedge n(N,G) = 0} N$$

In particular, it follows that

$$\bigcup_{N \in \mathcal{F}^+(E)} N \in \{Q \in \mathcal{Q} \mid G \subseteq Q\}$$

$$\bigcup_{N \in \mathcal{F}^-(E)} N \in \{Q \in \mathcal{Q} \mid Q \subseteq G\}$$

Let $N \in \mathcal{F}^+(E)$. Let $y \in G$ such that $y \in N$ and $y \notin \bigcup_{N' \in ch(N)} N'$ (such a point y exists as $p^*(N, G) \neq 0$). Then, $N = \min_{\subseteq} \mathcal{K}_y$, and since $y \in G^+$, we must have $N \subseteq G^+$. Consequently, we have $\bigcup_{N \in \mathcal{F}^+(E)} N \subseteq G^+$, and then $\bigcup_{N \in \mathcal{F}^+(E)} N = G^+$ and $\mathcal{F}^+(E) = C[G^+]$.

Let $x \in G^- \setminus \bigcup_{N \in \mathcal{F}^-(E)} N$. Then, there exists $N \in \mathcal{K}_x$ such that $N \subseteq G^-$. As $x \notin \bigcup_{N \in \mathcal{F}^-(E)} N$, we have $N \notin \mathcal{F}^-(E)$, and in particular, $n(N, G) \neq 0$. But then, there exists $y \in N$ such that $y \notin G$, and thus, $G^- \not\subseteq G$: contradiction. Consequently, we have $G^- \subseteq \bigcup_{N \in \mathcal{F}^-(E)} N$, and then $G^- = \bigcup_{N \in \mathcal{F}^-(E)} N$ and $\mathcal{F}^-(E) = C[G^-]$. □

Proof of Property 15 From the definition of $\mathcal{F}^\sigma(E)$, it is easily proved that each node is processed at most once. For each one of these $O(|\mathcal{K}|)$ processed nodes, one equality (related to $p^*(N, G)$ or $n(N, G)$, which are assumed to be precomputed, see Remark 3) is tested, and the status of the node ("in" or "out of" the result $\mathcal{F}^\sigma(E)$) is possibly modified. These two operations have a constant algorithmic complexity $O(1)$. The whole process then presents a linear complexity $O(|\mathcal{K}|)$. The generation of G^σ from $\mathcal{F}^\sigma(E)$ can be performed by modifying, for each node N of $\mathcal{K}$ and for each point x of N (these points being stored in E_N for each node N, see Remark 3) the status of x to indicate that it belongs to G^σ. This process then presents an algorithmic complexity $O(|E|)$. Hence the result holds. □

Proof of Property 19 Let us suppose that $ch(E) = \emptyset$. Then we have $\mathcal{Q} = \{\emptyset, E\}$, $d^\alpha(\emptyset, G) = (1-\alpha).p(E,G)$ and $d^\alpha(E,G) = \alpha.n(E,G)$. If we have

$$\alpha.n(E,G) < (1-\alpha).p^*(E,G) + \underbrace{\sum_{N \in ch(E)} c^\alpha(N)}_{=0}$$

then it comes $\mathcal{F}^\alpha(E) = \{E\}$, $c^\alpha(E) = \alpha.n(E)$ and thus we have

$$d^\alpha(G^\alpha, G) = d^\alpha(E,G) = c^\alpha(E) = \min_{Q \in \mathcal{Q}}\{d^\alpha(Q,G)\}$$

If we have

$$\alpha.n(E,G) \not< (1-\alpha).p(E,G)$$

then it comes $\mathcal{F}^\alpha(E) = \emptyset$, $c^\alpha(E) = (1-\alpha).p(E,G) = (1-\alpha).p^*(E,G)$ and thus we have

$$d^\alpha(G^\alpha, G) = d^\alpha(\emptyset, G) = c^\alpha(E) = \min_{Q \in \mathcal{Q}}\{d^\alpha(Q,G)\}$$

Consequently, the property is true whenever $ch(E) = \emptyset$.

Let us now suppose that $ch(E) \neq \emptyset$ and that the property holds for any $N \in ch(E)$ (with respect to $I_{|N}$, T_N and $G \cap N$, instead of I, T and G, see Property 5). Note that

$$\min_{Q \in \mathcal{Q}}\{d^\alpha(Q,G)\} = \min\{d^\alpha(E,G), \min_{Q \in \mathcal{Q}\setminus\{E\}}\{d^\alpha(Q,G)\}\}$$

while

$$d^\alpha(E,G) = \alpha.|E \setminus G| = \alpha.n(E)$$

and

$$\min_{Q \in \mathcal{Q}\setminus\{E\}}\{d^\alpha(Q,G)\} = \min_{Q \in \mathcal{Q}\setminus\{E\}} \alpha.|Q \setminus G| + (1-\alpha).|G \setminus Q|$$

Note also that $\{Q \cap N\}_{N \in ch(E)}$ is a partition of Q whenever $Q \neq E$ while $\{G \setminus \bigcup_{N \in ch(E)} N\} \cup \{G \cap N\}_{N \in ch(E)}$ is a partition of G (by omitting the possibly empty subsets). If $Q \neq E$, we have

$$d^\alpha(Q,G) = \alpha.|Q \setminus G| + (1-\alpha).|G \setminus Q|$$

with

$$\begin{array}{l} Q \setminus G = \bigcup_{N \in ch(E)} (Q \cap N) \setminus G \\ G \setminus Q = ((G \setminus \bigcup_{N \in ch(E)} N) \cup \bigcup_{N \in ch(E)} (G \cap N)) \setminus Q \end{array}$$

Then, we have

$$\begin{array}{l} |Q \setminus G| = \sum_{N \in ch(E)} |(Q \cap N) \setminus G| \\ |G \setminus Q| = |(G \setminus \bigcup_{N \in ch(E)} N) \setminus Q| + \sum_{N \in ch(E)} |(G \cap N) \setminus Q| \end{array}$$

and thus

$$\begin{array}{l} \alpha.|Q \setminus G| + (1-\alpha).|G \setminus Q| \\ = \left\{ \begin{array}{l} \sum_{N \in ch(E)} \alpha.|(Q \cap N) \setminus G| \\ + \sum_{N \in ch(E)} (1-\alpha).|(G \cap N) \setminus Q| \\ + (1-\alpha).|(G \setminus \bigcup_{N \in ch(E)} N) \setminus Q| \end{array} \right. \\ = \left\{ \begin{array}{l} \sum_{N \in ch(E)} \alpha.|(Q \cap N) \setminus (G \cap N)| \\ + \sum_{N \in ch(E)} (1-\alpha).|(G \cap N) \setminus (Q \cap N)| \\ + (1-\alpha). \underbrace{|G \setminus \bigcup_{N \in ch(E)} N|}_{=p^*(E)} \end{array} \right. \end{array}$$

From the above partition properties, it then comes that

$$\begin{array}{l}
\min_{Q\in\mathcal{Q}\setminus\{E\}}\{d^\alpha(Q,G)\}\\
= \min_{Q\in\mathcal{Q}\setminus\{E\}}\{\alpha.|Q\setminus G| + (1-\alpha).|G\setminus Q|\}\\
= \min_{Q\in\mathcal{Q}\setminus\{E\}}\left\{\begin{array}{l}\sum_{N\in ch(E)}\alpha.|(Q\cap N)\setminus(G\cap N)|\\ +\sum_{N\in ch(E)}(1-\alpha).|(G\cap N)\setminus(Q\cap N)|\\ +(1-\alpha).p^*(E)\end{array}\right\}\\
= (1-\alpha).p^*(E) + \sum_{N\in ch(E)} d^\alpha(Q\cap N, G\cap N)\\
= (1-\alpha).p^*(E) + \sum_{N\in ch(E)} c^\alpha(N)
\end{array}$$

by induction hypothesis. Consequently, we have

$$\min_{Q\in\mathcal{Q}}\{d^\alpha(Q,G)\} = \min\{\alpha.n(E), (1-\alpha).p^*(E) + \sum_{N\in ch(E)} c^\alpha(N)\}$$

and the result follows by induction from Definition 16. □

Proof of Property 20 The proof is similar to the proof of Property 15. The only difference lies in the fact that the set of conditions to be tested ($\alpha.n(N) \prec (1-\alpha).p^*(N) + \sum_{N'\in ch(N)} c^\alpha(N')$) requires at most $|\mathcal{K}|$ comparison operations ($\prec$) and $4.|\mathcal{K}|$ arithmetic operations ($.,+,-$), while the computation of all the terms $c^\alpha(.)$ involves (at most) the value $c^\alpha(N')$ only once for any $N'\in\mathcal{K}$, leading to less than $|\mathcal{K}|$ additions in the set of all the $\sum$ terms. Such supplementary operations then do not increase the algorithmic complexity $O(|\mathcal{K}|)$ of the computation of $\mathcal{F}^\alpha(E)$ by comparison to $\mathcal{F}^\sigma(E)$. Hence the result holds. □

Proof of Property 22 Let $N_1\in\mathcal{F}^{\alpha_1}$ and $N_2\in\mathcal{F}^{\alpha_2}$. By definition, $N_1\in C[G^{\alpha_1}]$ and $N_2\in C[G^{\alpha_2}]$ are connected components of G^{α_1} and G^{α_2}, respectively, which verify either $(N_1\subseteq N_2)\vee(N_2\subseteq N_1)$ or $(N_1\cap N_2=\emptyset)$ (see Property 8). Let us suppose that $G^{\alpha_2}\not\subseteq G^{\alpha_1}$. Then, there exists a node $N\in\mathcal{K}$ such that $N\in\mathcal{F}^{\alpha_2}$ while $\forall N'\in\mathcal{F}^{\alpha_1}, N\not\subseteq N'$. Let $\mathcal{K}'\subseteq\mathcal{F}^{\alpha_1}$ be the set of nodes forming the part of G^{α_1} included in N, *i.e.*, such that $\bigcup_{N'\in\mathcal{K}'} N' = N\cap G^{\alpha_1}$ (note that we may possibly have $\mathcal{K}'=\emptyset$). Let G'^{α_2} be the set defined by $G'^{\alpha_2} = (G^{\alpha_2}\setminus N)\cup(N\cap G^{\alpha_1})$, *i.e.*, the set obtained by substituting the nodes of $\mathcal{K}'$ to the node N in G^{α_2}. Let $t = |N\cap G|$, $f = |N\setminus G|$, $t' = |(N\cap G^{\alpha_1})\cap G|$ and $f' = |(N\cap G^{\alpha_1})\setminus G|$, with, obviously, $f'<f$ and $t'<t$. Then, we have

$$\begin{array}{l}
d^{\alpha_2}(G^{\alpha_2},G) = d^{\alpha_2}(G^{\alpha_2}\setminus N, G) + \alpha_2.f - (1-\alpha_2).t\\
d^{\alpha_2}(G'^{\alpha_2},G) = d^{\alpha_2}(G^{\alpha_2}\setminus N, G) + \alpha_2.f' - (1-\alpha_2).t'
\end{array}$$

Moreover, from the very definition of G^{α_2}, we have

$$d^{\alpha_2}(G^{\alpha_2},G) \le d^{\alpha_2}(G'^{\alpha_2},G)$$

and then, it comes

$$\alpha_2.(f-f') \le (1-\alpha_2).(t-t')$$

Now, let G'^{α_1} be the set defined by $G'^{\alpha_1} = G^{\alpha_1}\cup N$, *i.e.*, the set obtained by substituting the node N to the nodes of $\mathcal{K}'$ in G^{α_1}. We have

$$\begin{array}{l}
d^{\alpha_1}(G^{\alpha_1},G) = d^{\alpha_1}(G^{\alpha_1}\setminus N, G) + \alpha_1.f' - (1-\alpha_1).t'\\
d^{\alpha_1}(G'^{\alpha_1},G) = d^{\alpha_1}(G^{\alpha_1}\setminus N, G) + \alpha_1.f - (1-\alpha_1).t
\end{array}$$

Moreover, from the very definition of G^{α_1}, we have

$$d^{\alpha_1}(G^{\alpha_1},G) \le d^{\alpha_1}(G'^{\alpha_1},G)$$

and then, it comes

$$(1-\alpha_1).(t-t') \le \alpha_1.(f-f')$$

But in such conditions, we have

$$\alpha_2.(f-f') \le (1-\alpha_2).(t-t') \le (1-\alpha_1).(t-t') \le \alpha_1.(f-f')$$

which straightforwardly implies that $f-f'=t-t'=0$, a contradiction. Consequently, we have $G^{\alpha_1}\subseteq G^{\alpha_2}$. □

References

[1] P. Hanusse, P. Guillataud, Sémantique des images par analyse dendronique, in: Reconnaissance des Formes et Intelligence Artificielle - RFIA 1991, Proceedings, Vol. 2, 1991, pp. 577–588.

[2] L. Chen, M. W. Berry, W. W. Hargrove, Using dendronal signatures for feature extraction and retrieval, International Journal of Imaging Systems and Technology 11 (4) (2000) 243–253.

[3] J. Mattes, J. Demongeot, Efficient algorithms to implement the confinement tree, in: Discrete Geometry for Computer Imagery - DGCI 2000, Proceedings, Vol. 1953 of Lecture Notes in Computer Science, Springer, 2000, pp. 392–405.

[4] P. Salembier, A. Oliveras, L. Garrido, Anti-extensive connected operators for image and sequence processing, IEEE Transactions on Image Processing 7 (4) (1998) 555–570.

[5] D. Wishart, Mode analysis: A generalization of the nearest neighbor, in: A. J. Cole (Ed.), Numerical Taxonomy, Academic Press, London, 1969, pp. 282–319.

[6] J. A. Hartigan, Statistical theory in clustering, Journal of Classification 2 (1) (1985) 63–76.

[7] E. J. Breen, R. Jones, Attribute openings, thinnings, and granulometries, Computer Vision and Image Understanding 64 (3) (1996) 377–389.

[8] P. Salembier, Connected operators based on tree pruning strategies, in: L. Najman, H. Talbot (Eds.), Mathematical morphology: from theory to applications, ISTE/J. Wiley & Sons, 2010, Ch. 7, pp. 179–198.

[9] B. Naegel, N. Passat, Component-trees and multivalued images: A comparative study, in: International Symposium on Mathematical Morphology - ISMM 2009, Proceedings, Vol. 5720 of Lecture Notes in Computer Science, Springer, 2009, pp. 261–271.

[10] N. Passat, B. Naegel, An extension of component-trees to partial orders, in: International Conference on Image Processing - ICIP 2009, Proceedings, IEEE Signal Processing Society, 2009, pp. 3981–3984.

[11] G. Palma, I. Bloch, S. Muller, Fast fuzzy connected filter implementation using max-tree updates, Fuzzy Sets and Systems 161 (1) (2010) 118–146.

[12] E. R. Urbach, N. J. Boersma, M. H. F. Wilkinson, Vector attribute filters, in: International Symposium on Mathematical Morphology - ISMM 2005, Proceedings, Vol. 30 of Computational Imaging and Vision, Springer SBM, 2005, pp. 95–104.

[13] B. Caldairou, B. Naegel, N. Passat, Segmentation of complex images based on component-trees: Methodological tools, in: International Symposium on Mathematical Morphology - ISMM 2009, Proceedings, Vol. 5720 of Lecture Notes in Computer Science, Springer, 2009, pp. 171–180.

[14] B. Naegel, L. Wendling, Combining shape descriptors and component-tree for recognition of ancient graphical drop caps, in: Computer Vision Theory and Applications - VISAPP 2009, Proceedings, Vol. 2, 2009, pp. 297–302.

[15] L. Najman, M. Couprie, Building the component tree in quasi-linear time, IEEE Transactions on Image Processing 15 (11) (2006) 3531–3539.

[16] D. Menotti, L. Najman, A. de Albuquerque Araújo, 1D component tree in linear time and space and its application to gray-level image multithresholding, in: International Symposium on Mathematical Morphology - ISMM 2007, Proceedings, Vol. 1, INPE, 2007, pp. 437–448.

[17] C. Berger, T. Géraud, R. Levillain, N. Widynski, A. Baillard, E. Bertin, Effective component tree computation with application to pattern recognition in astronomical imaging, in: International Conference on Image Processing - ICIP 2007, Proceedings, IEEE Signal Processing Society, 2007, pp. 41–44.

[18] M. H. F. Wilkinson, H. Gao, W. H. Hesselink, J.-E. Jonker, A. Meijster, Concurrent computation of attribute filters on shared memory parallel machines, IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (10) (2008) 1800–1813.

[19] M. H. F. Wilkinson, M. A. Westenberg, Shape preserving filament enhancement filtering, in: Medical Image Computing and Computer-Assisted Intervention - MICCAI 2001, Proceedings, Vol. 2208 of Lecture Notes in Computer Science, Springer, 2001, pp. 770–777.

[20] E. R. Urbach, M. H. F. Wilkinson, Shape-only granulometries and gray-scale shape filters, in: International Symposium on Mathematical Morphology - ISMM 2002, Proceedings, CSIRO Publishing, 2002, pp. 305–314.

[21] C. Caldairou, N. Passat, B. Naegel, Attribute-filtering and knowledge extraction for vessel segmentation, in: International Symposium on Visual Computing - ISVC 2010, Proceedings, Vol. 6453 of Lecture Notes in Computer Science, Springer, 2010, pp. 13–22.

[22] P. Dokládal, I. Bloch, M. Couprie, D. Ruijters, R. Urtasun, L. Garnero, Topologically controlled segmentation of 3D magnetic resonance images of the head by using morphological operators, Pattern Recognition 36 (10) (2003) 2463–2478.

[23] B. Naegel, N. Passat, N. Boch, M. Kocher, Segmentation using vector-attribute filters: methodology and application to dermatological imaging, in: International Symposium on Mathematical Morphology - ISMM 2007, Proceedings, Vol. 1, INPE, 2007, pp. 239–250.

[24] G. K. Ouzounis, M. H. F. Wilkinson, Mask-based second-generation connectivity and attribute filters, IEEE Transactions on Pattern Analysis and Machine Intelligence 29 (6) (2007) 990–1004.

[25] R. Jones, Connected filtering and segmentation using component trees, Computer Vision and Image Understanding 75 (3) (1999) 215–228.

[26] J. Mattes, M. Richard, J. Demongeot, Tree representation for image matching and object recognition, in: Discrete Geometry for Computer Imagery - DGCI 1999, Proceedings, Vol. 1568 of Lecture Notes in Computer Science, Springer, 1999, pp. 298–312.

[27] V. Mosorov, A main stem concept for image matching, Pattern Recognition Letters 26 (8) (2005) 1105–1117.

[28] N. Alajlan, M. S. Kamel, G. H. Freeman, Geometry-based image retrieval in binary image databases, IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (6) (2008) 1003–1013.

[29] E. R. Urbach, J. B. T. M. Roerdink, M. H. F. Wilkinson, Connected shape-size pattern spectra for rotation and scale-invariant classification of gray-scale images, IEEE Transactions on Pattern Analysis and Machine Intelligence 29 (2) (2007) 272–285.

[30] M. A. Westenberg, J. B. T. M. Roerdink, M. H. F. Wilkinson, Volumetric attribute filtering and interactive visualization using the max-tree representation, IEEE Transactions on Image Processing 16 (12) (2007) 2943–2952.

[31] B. Naegel, L. Wendling, A document binarization method based on connected operators, Pattern Recognition Letters 31 (11) (2010) 1251–1259.

[32] M. Kass, A. Witkin, D. Terzopoulos, Snakes: Active contour models, International Journal of Computer Vision 1 (4) (1989) 321–331.

[33] J. A. Sethian, Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science, Cambridge University Press, 1999.

[34] W. A. Barrett, E. N. Mortensen, Interactive live-wire boundary extraction, Medical Image Analysis 1 (4) (1997) 331–341.

[35] E. N. Mortensen, W. A. Barrett, Interactive segmentation with intelligent scissors, Graphical Models and Image Processing 60 (5) (1998) 349–384.
[36] L. Vincent, P. Soille, Watersheds in digital spaces: an efficient algorithm based on immersion simulations, IEEE Transactions on Pattern Analysis and Machine Intelligence 13 (6) (1991) 583–598.
[37] G. Bertrand, M. Couprie, J. Cousty, L. Najman, Watersheds in discrete spaces, in: L. Najman, H. Talbot (Eds.), Mathematical morphology: from theory to applications, ISTE/J. Wiley & Sons, 2010, Ch. 3, pp. 81–107.
[38] R. Adams, L. Bischof, Seeded region growing, IEEE Transactions on Pattern Analysis and Machine Intelligence 16 (6) (1994) 641–647.
[39] A. Mehnert, P. Jackway, An improved seeded region growing algorithm, Pattern Recognition Letters 18 (10) (1997) 1065–1071.
[40] Y. Boykov, M. Jolly, Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images, in: International Conference on Computer Vision - ICCV 2001, Proceedings, Vol. 1, 2001, pp. 105–112.
[41] Y. Boykov, G. Funka-Lea, Graph cuts and efficient N-D image segmentation, International Journal on Computer Vision 70 (2) (2006) 109–131.
[42] P. Salembier, L. Garrido, Binary partition tree as an efficient representation for image processing, segmentation and information retrieval, IEEE Transactions on Image Processing 9 (4) (2000) 561–576.
[43] T. Adamek, Using contour information and segmentation for object registration, modeling and retrieval, Ph.D. thesis, Dublin City University (2006).
[44] K. McGuinness, N. E. O'Connor, A comparative evaluation of interactive segmentation algorithms, Pattern Recognition 43 (2) (2010) 434–444.
[45] D. Martin, C. Fowlkes, D. Tal, J. Malik, A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics, in: International Conference on Computer Vision - ICCV 2001, Proceedings, Vol. 2, 2001, pp. 416–423.
[46] C. A. Cocosco, V. Kollokian, R. K.-S. Kwan, A. C. Evans, BrainWeb: Online interface to a 3D MRI simulated brain database, in: Human Brain Mapping - HBM 1997, Proceedings, Vol. 5(4 Pt 2) of NeuroImage, 1997, p. S425.
[47] O. A. Glenn, A. J. Barkovich, Magnetic resonance imaging of the fetal brain and spine: An increasingly important tool in prenatal diagnosis, part 1, American Journal of Neuroradiology 27 (8) (2006) 1604–1611.
[48] S. M. Smith, Fast robust automated brain extraction, Human Brain Mapping 17 (3) (2002) 143–155.
[49] B. Dogdas, D. W. Shattuck, R. M. Leahy, Segmentation of skull and scalp in 3-D human MRI using mathematical morphology, Human Brain Mapping 26 (4) (2005) 273–285.
[50] S. A. Sadananthan, W. Zheng, M. W. L. Chee, V. Zagorodnov, Skull stripping using graph cuts, NeuroImage 49 (1) (2010) 225–239.
[51] L. Ford, D. Fulkerson, Flows in networks, Princeton University Press, 1962.
[52] Y. Boykov, V. Kolmogorov, An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (9) (2004) 1124–1137.
[53] J. Serra, Connectivity on complete lattices, Journal of Mathematical Imaging and Vision 9 (3) (1998) 231–251.
[54] P. Monasse, F. Guichard, Scale-space from a level lines tree, Journal of Visual Communication and Image Representation 11 (2) (2000) 224–236.